

9.3

*IBM MQ -Guide de référence des  
applications de développement*

**IBM**

**Remarque**

Avant d'utiliser le présent document et le produit associé, prenez connaissance des informations générales figurant à la section [«Remarques»](#), à la page 2297.

Cette édition s'applique à la version 9 édition 3 d' IBM® MQ et à toutes les éditions et modifications ultérieures, sauf indication contraire dans les nouvelles éditions.

Lorsque vous envoyez des informations à IBM, vous accordez à IBM le droit non exclusif d'utiliser ou de distribuer les informations de la manière qu'il juge appropriée, sans aucune obligation de votre part.

© **Copyright International Business Machines Corporation 2007, 2024.**

---

# Table des matières

<b>Références relatives au développement d'applications.....</b>	<b>7</b>
Référence des applications MQI.....	7
Exemples de code.....	8
Constantes.....	61
Types de données utilisés dans l'interface MQI.....	237
Appels de fonctions.....	654
Attributs des objets.....	835
Codes retour.....	917
Règles de validation des options MQI.....	918
Messages de commande de publication / abonnement en file d'attente.....	921
Codages de machine.....	944
Options de rapport et indicateurs de message.....	947
Exit de conversion de données.....	951
Propriétés spécifiées en tant qu'éléments MQRFH2.....	975
Conversion de page de code.....	984
Normes de codage sur les plateformes 64 bits.....	1038
IBM i Application Programming Reference (ILE/RPG).....	1042
Description des types de données sous IBM i.....	1043
Appels de fonction sur IBM i.....	1311
Attributs des objets sur IBM i.....	1435
Applications.....	1484
Codes retour pour IBM i (ILE RPG).....	1498
Règles de validation des options MQI pour IBM i (ILE RPG).....	1499
Codages de machine sur IBM i.....	1502
Options de rapport et indicateurs de message sous IBM i.....	1505
Conversion de données sous IBM i.....	1508
Traitement de la conversion sous IBM i.....	1509
Conventions de traitement sous IBM i.....	1510
Conversion des messages de rapport sur IBM i.....	1514
MQDXP (paramètre d'exit de conversion de données) sous IBM i.....	1515
MQXCNCV (conversion de caractères) sous IBM i.....	1520
MQCONVX (exit de conversion de données) sous IBM i.....	1525
Références des exits utilisateur, des exits API et des services installables.....	1529
Structure MQIEP.....	1529
Référence de l'exit de conversion de données.....	1533
MQ_PUBLISH_EXIT-Exit de publication.....	1537
Structures de données et appels d'exit de canal.....	1545
Appel d'exit de charge de travail de cluster et structures de données.....	1611
Référence d'exit API.....	1638
Informations de référence de l'interface des services installables.....	1699
Informations de référence de l'interface des services installables sur IBM i.....	1764
Les classes et interfaces IBM MQ .NET.....	1804
Classe MQAsyncStatus.NET.....	1804
Classe MQAuthenticationInformationRecord.NET.....	1805
Classe MQDestination.NET.....	1806
Classe MQEnvironment.NET.....	1809
Classe MQException.NET.....	1812
Classe MQGetMessageOptions.NET.....	1812
Classe MQManagedObject.NET.....	1815
Classe MQMessage.NET.....	1818
Classe MQProcess.NET.....	1830
Classe MQPropertyDescriptor.NET.....	1833

Classe MQPutMessageOptions.NET.....	1834
Classe MQQueue.NET.....	1837
Classe MQQueueManager.NET.....	1845
Classe MQSubscription.NET.....	1858
Classe MQTopic.NET.....	1860
Interface IMQObjectTrigger.NET.....	1866
Interface MQC.NET.....	1866
Identificateurs de jeu de caractères pour les applications .NET.....	1866
IBM MQ classes C++.....	1869
Références croisées C++ et MQI.....	1871
Classe C++ d'enregistrement ImqAuthentication.....	1888
Classe C++ ImqBinary.....	1890
Classe C++ ImqCache.....	1892
Classe C++ ImqChannel.....	1895
Classe C++ imqCICSBridgeHeader.....	1901
Classe C++ ImqDeadLetterHeader.....	1907
ImqDistributionRépertoire la classe C++.....	1910
Classe C++ ImqError.....	1911
Classe C++ ImqGetMessageOptions.....	1912
Classe C++ ImqHeader.....	1916
Classe C++ imqIMSBridgeHeader.....	1917
Classe C++ ImqItem.....	1920
Classe C++ ImqMessage.....	1922
Classe C++ de la fonction de suivi ImqMessage.....	1929
Classe C++ ImqNamelist.....	1932
Classe C++ ImqObject.....	1933
Classe C++ ImqProcess.....	1939
Classe C++ ImqPutMessageOptions.....	1941
Classe C++ ImqQueue.....	1943
Classe C++ du gestionnaire ImqQueue.....	1954
Classe C++ d'en-tête ImqReference.....	1971
Classe C++ ImqString.....	1973
Classe C++ ImqTrigger.....	1979
Classe C++ d'en-tête ImqWork.....	1981
Propriétés des objets IBM MQ classes for JMS.....	1983
Dépendances entre les propriétés des objets IBM MQ classes for JMS.....	1987
APPLICATIONNAME.....	1989
ASYNCEXCEPTION.....	1990
OPTIONS balance.....	1991
TYPE D'ÉQUILIBRAGE.....	1991
DÉLAI_ATTENTE.....	1992
BROKERCCDURSUBQ.....	1992
BROKERCCSUBQ.....	1993
BROKERCONQ.....	1993
BROKERDURSUBQ.....	1994
BROKERPUBQ.....	1994
BROKERPUBQMGR.....	1995
BROKERQMGR.....	1995
BROKERSUBQ.....	1995
BROKERVER.....	1996
CCDTURL.....	1997
CCSID.....	1997
Canal.....	1998
CLEANUP.....	1998
CLEANUPINT.....	1999
ConnectionNameList.....	1999
CLIENTRECONNECTOPTIONS.....	1999
CLIENTRECONNECTTIMEOUT.....	2000



IDCLIENT.....	2001
CLONESUPP.....	2001
COMPHDR.....	2002
COMPMSG.....	2002
CONNOPT.....	2003
CONNTAG.....	2004
DESCRIPTION.....	2004
DIRECTAUTH.....	2005
ENCODING.....	2005
EXPIRY.....	2006
FAILIFQUIESCE.....	2007
HOSTNAME.....	2007
LOCALADDRESS.....	2008
NOM_MAPPE.....	2008
MAXBUFFSIZE.....	2009
MDREAD.....	2009
MDWRITE.....	2010
MDMSGCTX.....	2011
MSGBATCHSZ.....	2011
MSGBODY.....	2012
MSGRETENTION.....	2012
MSGSELECTION.....	2013
MULTICAST.....	2013
OPTIMISTICPUBLICATION.....	2014
OUTCOMENOTIFICATION.....	2015
PERSISTENCE.....	2015
POLLINGINT.....	2016
PORT.....	2016
PRIORITY.....	2017
PROCESSDURATION.....	2017
PROVIDERVERSION.....	2018
PROXYHOSTNAME.....	2020
PROXYPORT.....	2021
PUBACKINT.....	2021
PUTASYNCALLOWED.....	2021
QMANAGER.....	2022
QUEUE.....	2023
READAHEADALLOWED.....	2023
READAHEADCLOSEPOLICY.....	2024
RECEIVECCSID.....	2024
RECEIVECONVERSION.....	2025
RECEIVEISOLATION.....	2025
RECEXIT.....	2026
RECEXITINIT.....	2026
REPLYTOSTYLE.....	2027
RESCANINT.....	2027
SECEXIT.....	2028
SECEXITINIT.....	2028
SENDCHECKCOUNT.....	2029
SENDEXIT.....	2029
SENDEXITINIT.....	2030
SHARECONVALLOWED.....	2030
SPARSESUBS.....	2031
SSLCIPHERSUITE.....	2032
SSLCRL.....	2032
SSLFIPSREQUIRED.....	2032
SSLPEERNAME.....	2033
SSLRESETCOUNT.....	2033

STATREFRESHINT.....	2034
SUBSTORE.....	2034
SYNCPOINTALLGETS.....	2035
TARGCLIENT.....	2035
TARGCLIENTMATCHING.....	2036
TEMPMODEL.....	2036
TEMPQPREFIX.....	2037
TEMPTOPICPREFIX.....	2037
TOPIC.....	2038
TRANSPORT.....	2038
WILDCARDFORMAT.....	2039
Propriété ENCODING.....	2039
Propriétés TLS des objets JMS.....	2040
Référence IBM MQ Message Service Client (XMS) for .NET.....	2041
.NET interfaces.....	2041
Propriétés des objets XMS.....	2123
Managed File Transfer Référence de développement de d'applications.....	2192
Exemples d'utilisation de fteCreateTransfer pour démarrer des programmes.....	2192
<b>fteAnt</b> : exécuter les tâches Ant dans MFT.....	2194
Exits utilisateur MFT pour la référence de personnalisation.....	2220
Formats de message pour les messages que vous pouvez placer dans la file d'attente de commandes de l'agent MFT.....	2262
Informations de référence sur la messagerie REST API.....	2262
ressources REST API.....	2262
<b>Remarques.....</b>	<b>2297</b>
Documentation sur l'interface de programmation.....	2298
Marques.....	2299

# Références relatives au développement d'applications

---

Utilisez les liens fournis dans cette section pour vous aider à développer vos applications IBM MQ .

- [«Référence des applications MQI», à la page 7](#)
-  [«IBM i Application Programming Reference \(ILE/RPG\)», à la page 1042](#)
-  [«Conversion de données sous IBM i», à la page 1508](#)
- [«Références des exits utilisateur, des exits API et des services installables», à la page 1529](#)
- [«Les classes et interfaces IBM MQ .NET», à la page 1804](#)
- [«IBM MQ classes C++», à la page 1869](#)
- [«Propriétés des objets IBM MQ classes for JMS», à la page 1983](#)
- [«Informations de référence sur la messagerie REST API», à la page 2262](#)

## Tâches associées

[Développement d'applications](#)

## Référence associée

[Classes IBM MQ pour les bibliothèques Java](#)

[IBM MQ classes for JMS](#)

## Référence des applications MQI

---

Utilisez les liens fournis dans cette section pour vous aider à développer vos applications MQI (Message Queue Interface).

- [«Exemples de code», à la page 8](#)
- [«Constantes», à la page 61](#)
- [«Types de données utilisés dans l'interface MQI», à la page 237](#)
- [«Appels de fonctions», à la page 654](#)
- [«Attributs des objets», à la page 835](#)
- [«Codes retour», à la page 917](#)
- [«Règles de validation des options MQI», à la page 918](#)
- [«Codages de machine», à la page 944](#)
- [«Options de rapport et indicateurs de message», à la page 947](#)
- [«Exit de conversion de données», à la page 951](#)
- [«Propriétés spécifiées en tant qu'éléments MQRFH2», à la page 975](#)
- [«Conversion de page de code», à la page 984](#)

## Concepts associés

[«Références des exits utilisateur, des exits API et des services installables», à la page 1529](#)

Utilisez les informations de la présente section pour vous aider à développer vos exits utilisateur, vos exits API et vos applications de services installables:

## Tâches associées

[Développement d'applications](#)

## Référence associée

[«Les classes et interfaces IBM MQ .NET», à la page 1804](#)

Les classes et les interfaces IBM MQ .NET sont répertoriées par ordre alphabétique. Les propriétés, les méthodes et les constructeurs sont décrits.

«IBM MQ classes C++», à la page 1869

Les classes C++ IBM MQ encapsulent l'interface MQI ( IBM MQ Message Queue Interface). Il existe un fichier d'en-tête C++ unique, **imqi.hpp**, qui couvre toutes ces classes.

Les bibliothèques IBM MQ Classes for Java

IBM MQ Classes pour JMS

## Exemples de code

Utilisez les informations de référence présentées dans cette section pour accomplir les tâches adaptées à vos besoins métier.

## Exemples de langage C

Cette collection de rubriques est principalement extraite des modèles d'application IBM MQ for z/OS . Elles sont applicables à toutes les plateformes, sauf indication contraire.

### ***Connexion à un gestionnaire de files d'attente***

Cet exemple montre comment utiliser l'appel MQCONN pour connecter un programme à un gestionnaire de files d'attente dans z/OS batch.

Cette extraction est extraite de l'exemple d'application Parcourir (programme CSQ4BCA1) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```
#include <cmqc.h>
:
static char Parm1[MQ_Q_MGR_NAME_LENGTH] ;

int main(int argc, char *argv[] )
{
    /*                                     */
    /* Variables for MQ calls             */
    /*                                     */
    MQHCONN Hconn;      /* Connection handle */
    MQLONG  CompCode;   /* Completion code  */
    MQLONG  Reason;     /* Qualifying reason */

    /* Copy the queue manager name, passed in the */
    /* parm field, to Parm1                        */
    strncpy(Parm1,argv[1],MQ_Q_MGR_NAME_LENGTH);

    /*                                     */
    /* Connect to the specified queue manager.     */
    /* Test the output of the connect call. If the */
    /* call fails, print an error message showing the */
    /* completion code and reason code, then leave the */
    /* program.                                     */
    /*                                     */
    MQCONN(Parm1,
           &Hconn,
           &CompCode,
           &Reason);
    if ((CompCode != MQCC_OK) | (Reason != MQRC_NONE))
    {
        sprintf(pBuff, MESSAGE_4_E,
                ERROR_IN_MQCONN, CompCode, Reason);
        PrintLine(pBuff);
        RetCode = CSQ4_ERROR;
        goto AbnormalExit2;
    }
    :
}
```

### ***Déconnexion d'un gestionnaire de files d'attente***

Cet exemple montre comment utiliser l'appel MQDISC pour déconnecter un programme d'un gestionnaire de files d'attente dans un lot z/OS .



Les variables utilisées dans cette extraction de code sont celles qui ont été définies dans «Connexion à un gestionnaire de files d'attente», à la page 8. Cette extraction est extraite de l'exemple d'application Parcourir (programme CSQ4BCA1) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```

:
/*
/* Disconnect from the queue manager. Test the
/* output of the disconnect call. If the call
/* fails, print an error message showing the
/* completion code and reason code.
/*
MQDISC(&Hconn,
      &CompCode,
      &Reason);
if ((CompCode != MQCC_OK) || (Reason != MQRC_NONE))
{
  sprintf(pBuff, MESSAGE_4_E,
          ERROR_IN_MQDISC, CompCode, Reason);
  PrintLine(pBuff);
  RetCode = CSQ4_ERROR;
}
:

```

### ***Création d'une file d'attente dynamique***

Cet exemple montre comment utiliser l'appel MQOPEN pour créer une file d'attente dynamique.

Cet extrait est extrait de l'exemple d'application Mail Manager (programme CSQ4TCD1) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```

:
MQLONG HCONN = 0; /* Connection handle */
MQHOBJ HOBJ; /* MailQ Object handle */
MQHOBJ HobjTempQ; /* TempQ Object Handle */
MQLONG CompCode; /* Completion code */
MQLONG Reason; /* Qualifying reason */
MQOD ObjDesc = {MQOD_DEFAULT};
MQLONG OpenOptions; /* Options control MQOPEN */

/*-----*/
/* Initialize the Object Descriptor (MQOD)
/* control block. (The remaining fields
/* are already initialized.)
/*-----*/
strncpy( ObjDesc.ObjectName,
        SYSTEM_REPLY_MODEL,
        MQ_Q_NAME_LENGTH );
strncpy( ObjDesc.DynamicQName,
        SYSTEM_REPLY_INITIAL,
        MQ_Q_NAME_LENGTH );
OpenOptions = MQOO_INPUT_AS_Q_DEF;
/*-----*/
/* Open the model queue and, therefore,
/* create and open a temporary dynamic
/* queue
/*-----*/
MQOPEN( HCONN,
        &ObjDesc,
        OpenOptions,
        &HobjTempQ,
        &CompCode,
        &Reason );
if ( CompCode == MQCC_OK ) {
}
else {
  /*-----*/
  /* Build an error message to report the
  /* failure of the opening of the model
  /* queue
  /*-----*/
}

```

```

        MQMErrorHandling( "OPEN TEMPQ", CompCode,
                          Reason );
    }
    ErrorFound = TRUE;
}
return ErrorFound;
...

```

### ***Ouverture d'une file d'attente existante***

Cet exemple montre comment utiliser l'appel MQOPEN pour ouvrir une file d'attente qui a déjà été définie.

Cette extraction est extraite de l'exemple d'application Parcourir (programme CSQ4BCA1) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```

#include <cmqc.h>
...
static char Parm1[MQ_Q_MGR_NAME_LENGTH];
...
int main(int argc, char *argv[] )
{
    /*
     * Variables for MQ calls
     */
    /*
     * MQHCONN Hconn ;           /* Connection handle
     * MQLONG  CompCode;        /* Completion code
     * MQLONG  Reason;          /* Qualifying reason
     * MQOD    ObjDesc = { MQOD_DEFAULT };
     *                               /* Object descriptor
     * MQLONG  OpenOptions;     /* Options that control
     *                               /* the MQOPEN call
     * MQHOBJ  Hobj;           /* Object handle
     *
     * /* Copy the queue name, passed in the parm field,
     * /* to Parm2 strncpy(Parm2,argv[2],
     * /* MQ_Q_NAME_LENGTH);
     *
     * /*
     * /* Initialize the object descriptor (MQOD) control
     * /* block. (The initialization default sets StrucId,
     * /* Version, ObjectType, ObjectQMgrName,
     * /* DynamicQName, and AlternateUserid fields)
     * /*
     * strncpy(ObjDesc.ObjectName,Parm2,MQ_Q_NAME_LENGTH);
     *
     * /* Initialize the other fields required for the open
     * /* call (Hobj is set by the MQCONN call).
     * /*
     * OpenOptions = MQOO_BROWSE;
     *
     * /*
     * /* Open the queue.
     * /* Test the output of the open call. If the call
     * /* fails, print an error message showing the
     * /* completion code and reason code, then bypass
     * /* processing, disconnect and leave the program.
     * /*
     * MQOPEN(Hconn,
     *         &ObjDesc,
     *         OpenOptions,
     *         &Hobj,
     *         &CompCode,
     *         &Reason);
     *
     * if ((CompCode != MQCC_OK) || (Reason != MQRC_NONE))
     * {
     *     sprintf(pBuff, MESSAGE_4_E,
     *             ERROR_IN_MQOPEN, CompCode, Reason);
     *     PrintLine(pBuff);
     *     RetCode = CSQ4_ERROR;
     *     goto AbnormalExit1; /* disconnect processing */
     * }
     *
     * } /* end of main */

```

## Fermeture d'une file d'attente


Cet exemple montre comment utiliser l'appel MQCLOSE pour fermer une file d'attente.

Cette extraction est extraite de l'exemple d'application Parcourir (programme CSQ4BCA1) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```
:
/*                                     */
/* Close the queue.                    */
/* Test the output of the close call.  If the call */
/* fails, print an error message showing the */
/* completion code and reason code.      */
/*                                     */
MQCLOSE(Hconn,
        &Hobj,
        MQCO_NONE,
        &CompCode,
        &Reason);
if ((CompCode != MQCC_OK) || (Reason != MQRC_NONE))
{
    sprintf(pBuff, MESSAGE_4_E,
            ERROR_IN_MQCLOSE, CompCode, Reason);
    PrintLine(pBuff);
    RetCode = CSQ4_ERROR;
}
:
```

## Insertion d'un message à l'aide de MQPUT

Cet exemple montre comment utiliser l'appel MQPUT pour placer un message dans une file d'attente.

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ. Pour connaître les noms et les emplacements des exemples d'application, voir [Exemples de programmes procéduraux \(plateformes excepté z/OS\)](#)  et [Exemples de programmes pour IBM MQ for z/OS](#).

```
:
qput()
{
    MQMD    MsgDesc;
    MQPMO   PutMsgOpts;
    MQLONG  CompCode;
    MQLONG  Reason;
    MQHCONN Hconn;
    MQHOBJ  Hobj;
    char message_buffer[] = "MY MESSAGE";
    /*-----*/
    /* Set up PMO structure.                    */
    /*-----*/
    memset(&PutMsgOpts, '\0', sizeof(PutMsgOpts));
    memcpy(PutMsgOpts.StrucId, MQPMO_STRUC_ID,
           sizeof(PutMsgOpts.StrucId));
    PutMsgOpts.Version = MQPMO_VERSION_1;
    PutMsgOpts.Options = MQPMO_SYNCPOINT;

    /*-----*/
    /* Set up MD structure.                    */
    /*-----*/
    memset(&MsgDesc, '\0', sizeof(MsgDesc));
    memcpy(MsgDesc.StrucId, MQMD_STRUC_ID,
           sizeof(MsgDesc.StrucId));
    MsgDesc.Version      = MQMD_VERSION_1;
    MsgDesc.Expiry       = MQEI_UNLIMITED;
    MsgDesc.Report       = MQRO_NONE;
    MsgDesc.MsgType      = MQMT_DATAGRAM;
    MsgDesc.Priority     = 1;
    MsgDesc.Persistence  = MQPER_PERSISTENT;
    memset(MsgDesc.ReplyToQ,
           '\0',
           sizeof(MsgDesc.ReplyToQ));
    /*-----*/
    /* Put the message.                        */
    /*-----*/
    MQPUT(Hconn, Hobj, &MsgDesc, &PutMsgOpts,
```

```
sizeof(message_buffer), message_buffer,
&CompCode, &Reason);
```

```
/*-----*/
/* Check completion and reason codes. */
/*-----*/
switch (CompCode)
{
  case MQCC_OK:
    break;
  case MQCC_FAILED:
    switch (Reason)
    {
      case MQRC_Q_FULL:
      case MQRC_MSG_TOO_BIG_FOR_Q:
        break;
      default:
        break; /* Perform error processing */
    }
    break;
  default:
    break; /* Perform error processing */
}
}
```

### ***Insertion d'un message à l'aide de MQPUT1***

Cet exemple montre comment utiliser l'appel MQPUT1 pour ouvrir une file d'attente, insérer un message unique dans la file d'attente, puis fermer la file d'attente.

Cet extrait est extrait de l'exemple d'application Credit Check (programme CSQ4CCB5) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```
:
MQLONG Hconn; /* Connection handle */
MQHOBJ Hobj_CheckQ; /* Object handle */
MQLONG CompCode; /* Completion code */
MQLONG Reason; /* Qualifying reason */
MQOD ObjDesc = {MQOD_DEFAULT}; /* Object descriptor */
MQMD MsgDesc = {MQMD_DEFAULT}; /* Message descriptor */
MQLONG OpenOptions; /* Control the MQOPEN call */

MQGMO GetMsgOpts = {MQGMO_DEFAULT}; /* Get Message Options */
MQLONG MsgBufLen; /* Length of message buffer */
CSQ4BCAQ MsgBuffer; /* Message structure */
MQLONG DataLen; /* Length of message */

MQPMO PutMsgOpts = {MQPMO_DEFAULT}; /* Put Message Options */
CSQ4BQRM PutBuffer; /* Message structure */
MQLONG PutBufLen = sizeof(PutBuffer); /* Length of message buffer */
:
```

```
void Process_Query(void)
{
  /* Build the reply message */
  /* Set the object descriptor, message descriptor and
  /* put message options to the values required to
  /* create the reply message.
  /*
  strncpy(ObjDesc.ObjectName, MsgDesc.ReplyToQ,
          MQ_Q_NAME_LENGTH);
  strncpy(ObjDesc.ObjectQMgrName, MsgDesc.ReplyToQMgr,
          MQ_Q_MGR_NAME_LENGTH);
  MsgDesc.MsgType = MQMT_REPLY;
```

```

MsgDesc.Report = MQRO_NONE;
memset(MsgDesc.ReplyToQ, ' ', MQ_Q_NAME_LENGTH);
memset(MsgDesc.ReplyToQMGR, ' ', MQ_Q_MGR_NAME_LENGTH);
memcpy(MsgDesc.MsgId, MQMI_NONE, sizeof(MsgDesc.MsgId));
PutMsgOpts.Options = MQPMO_SYNCPOINT +
                    MQPMO_PASS_IDENTITY_CONTEXT;
PutMsgOpts.Context = Hobj_CheckQ;
PutBuffLen = sizeof(PutBuffer);
MQPUT1(Hconn,
        &ObjDesc,
        &MsgDesc,
        &PutMsgOpts,
        PutBuffLen,
        &PutBuffer,
        &CompCode,
        &Reason);

if (CompCode != MQCC_OK)
{
    strncpy(TS_Operation, "MQPUT1",
            sizeof(TS_Operation));
    strncpy(TS_ObjName, ObjDesc.ObjectName,
            MQ_Q_NAME_LENGTH);
    Record_Call_Error();
    Forward_Msg_To_DLQ();
}
return;
}
:

```

### ***obtention d'un message***

Cet exemple montre comment utiliser l'appel MQGET pour supprimer un message d'une file d'attente.

Cette extraction est extraite de l'exemple d'application Parcourir (programme CSQ4BCA1) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```

#include "cmqc.h"
:
#define BUFFERLENGTH 80
:
int main(int argc, char *argv[] )
{
    /*                                     */
    /*   Variables for MQ calls           */
    /*                                     */
    MQHCONN Hconn ;                       /* Connection handle   */
    MQLONG  CompCode;                      /* Completion code     */
    MQLONG  Reason;                        /* Qualifying reason   */
    MQHOBJ  Hobj;                          /* Object handle       */
    MQMD    MsgDesc = { MQMD_DEFAULT };    /* Message descriptor  */
    MQLONG  DataLength ;                   /* Length of the message */
    MQCHAR  Buffer[BUFFERLENGTH+1];        /* Area for message data */
    MQGMO   GetMsgOpts = { MQGMO_DEFAULT }; /* Options which control */
    /*                                     */
    /*   the MQGET call                   */
    MQLONG  BufferLength = BUFFERLENGTH ;   /* Length of buffer    */
    :
    /*   No need to change the message descriptor */
    /*   (MQMD) control block because initialization */
    /*   default sets all the fields.             */
    /*                                     */
    /*   Initialize the get message options (MQGMO) */
    /*   control block (the copy file initializes all */
    /*   the other fields).                       */
    /*                                     */
    GetMsgOpts.Options = MQGMO_NO_WAIT      +
                        MQGMO_BROWSE_FIRST +
                        MQGMO_ACCEPT_TRUNCATED_MSG;

    /*                                     */
    /* Get the first message.               */
    /* Test for the output of the call is carried out */
    /* in the 'for' loop.                   */
    /*                                     */
}

```

```

MQGET(Hconn,
      Hobj,
      &MsgDesc,
      &GetMsgOpts,
      BufferLength,
      Buffer,
      &DataLength,
      &CompCode,
      &Reason);

```

```

/*                                     */
/* Process the message and get the next message, */
/* until no messages remaining.                */
/*
/* If the call fails for any other reason, */
/* print an error message showing the completion */
/* code and reason code.                   */
/*
if ( (CompCode == MQCC_FAILED) &&
     (Reason == MQRC_NO_MSG_AVAILABLE) )
{
    ...
}
else
{
    sprintf(pBuff, MESSAGE_4_E,
            ERROR_IN_MQGET, CompCode, Reason);
    PrintLine(pBuff);
    RetCode = CSQ4_ERROR;
    ...
}
} /* end of main */

```

### **Obtention d'un message à l'aide de l'option d'attente**

Cet exemple montre comment utiliser l'option d'attente de l'appel MQGET.

Ce code accepte les messages tronqués. Cet extrait est extrait de l'exemple d'application Credit Check (programme CSQ4CCB5) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```

:
MQLONG  Hconn;           /* Connection handle      */
MQHOBJ  Hobj_CheckQ;    /* Object handle          */
MQLONG  CompCode;       /* Completion code        */
MQLONG  Reason;         /* Qualifying reason      */
MQOD    ObjDesc = {MQOD_DEFAULT};
/* Object descriptor      */
MQMD    MsgDesc = {MQMD_DEFAULT};
/* Message descriptor     */
MQLONG  OpenOptions;    /* Control the MQOPEN call */
MQGMO   GetMsgOpts = {MQGMO_DEFAULT};
/* Get Message Options    */
MQLONG  MsgBuffLen;     /* Length of message buffer */
CSQ4BCAQ MsgBuffer;     /* Message structure      */
MQLONG  DataLen;        /* Length of message      */

```

```

:
void main(void)
{
    :
    /*                                     */
    /* Initialize options and open the queue for input */
    /*
    :
    /*                                     */
    /* Get and process messages                */
    /*
    GetMsgOpts.Options = MQGMO_WAIT +
                        MQGMO_ACCEPT_TRUNCATED_MSG +
                        MQGMO_SYNCPOINT;
    GetMsgOpts.WaitInterval = WAIT_INTERVAL;

```

```

MsgBufLen = sizeof(MsgBuffer);
memcpy(MsgDesc.MsgId, MQMI_NONE,
        sizeof(MsgDesc.MsgId));
memcpy(MsgDesc.CorrelId, MQCI_NONE,
        sizeof(MsgDesc.CorrelId));

/*
/* Make the first MQGET call outside the loop
/*
MQGET(Hconn,
      Hobj_CheckQ,
      &MsgDesc,
      &GetMsgOpts,
      MsgBufLen,
      &MsgBuffer,
      &DataLen,
      &CompCode,
      &Reason);
:
/*
/* Test the output of the MQGET call. If the call
/* failed, send an error message showing the
/* completion code and reason code, unless the
/* reason code is NO_MSG_AVAILABLE.
/*
if (Reason != MQRC_NO_MSG_AVAILABLE)
{
  strncpy(TS_Operation, "MQGET", sizeof(TS_Operation));
  strncpy(TS_ObjName, ObjDesc.ObjectName,
          MQ_Q_NAME_LENGTH);
  Record_Call_Error();
}
:

```

## Obtention d'un message à l'aide de la notification

La signalisation est disponible uniquement avec IBM MQ for z/OS.

Cet exemple montre comment utiliser l'appel MQGET pour définir un signal afin d'être averti lorsqu'un message approprié arrive dans une file d'attente. Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```

:
get_set_signal()
{
  MQMD   MsgDesc;
  MQGMO  GetMsgOpts;
  MQLONG CompCode;
  MQLONG Reason;
  MQHCONN Hconn;
  MQHOBJ  Hobj;
  MQLONG  BufferLength;
  MQLONG  DataLength;
  char message_buffer[100];
  long int q_ecn, work_ecn;
  short int signal_sw, endloop;
  long int mask = 255;

  /*-----*/
  /* Set up GMO structure.
  /*-----*/
  memset(&GetMsgOpts, '\0', sizeof(GetMsgOpts));
  memcpy(GetMsgOpts.StrucId, MQGMO_STRUC_ID,
         sizeof(GetMsgOpts.StrucId));
  GetMsgOpts.Version      = MQGMO_VERSION_1;
  GetMsgOpts.WaitInterval = 1000;
  GetMsgOpts.Options      = MQGMO_SET_SIGNAL +
                           MQGMO_BROWSE_FIRST;

  q_ecn = 0;
  GetMsgOpts.Signal1 = &q_ecn;
  /*-----*/
  /* Set up MD structure.
  /*-----*/
  memset(&MsgDesc, '\0', sizeof(MsgDesc));
  memcpy(MsgDesc.StrucId, MQMD_STRUC_ID,
         sizeof(MsgDesc.StrucId));
  MsgDesc.Version = MQMD_VERSION_1;
  MsgDesc.Report  = MQRO_NONE;
  memcpy(MsgDesc.MsgId, MQMI_NONE,

```

```

        sizeof(MsgDesc.MsgId));
memcpy(MsgDesc.CorrelId, MQCI_NONE,
        sizeof(MsgDesc.CorrelId));

```

```

/*-----*/
/* Issue the MQGET call. */
/*-----*/
BufferLength = sizeof(message_buffer);
signal_sw = 0;

MQGET(Hconn, Hobj, &MsgDesc, &GetMsgOpts,
      BufferLength, message_buffer, &DataLength,
      &CompCode, &Reason);
/*-----*/
/* Check completion and reason codes. */
/*-----*/
switch (CompCode)
{
    case (MQCC_OK):          /* Message retrieved */
        break;
    case (MQCC_WARNING):
        switch (Reason)
        {
            case (MQRC_SIGNAL_REQUEST_ACCEPTED):
                signal_sw = 1;
                break;
            default:
                break; /* Perform error processing */
        }
        break;
    case (MQCC_FAILED):
        switch (Reason)
        {
            case (MQRC_Q_MGR_NOT_AVAILABLE):
            case (MQRC_CONNECTION_BROKEN):
            case (MQRC_Q_MGR_STOPPING):
                break;
            default:
                break; /* Perform error processing. */
        }
        break;
    default:
        break; /* Perform error processing. */
}
/*-----*/
/* If the SET_SIGNAL was accepted, set up a loop to */
/* check whether a message has arrived at one second */
/* intervals. The loop ends if a message arrives or */
/* the wait interval specified in the MQGMO */
/* structure has expired. */
/* If a message arrives on the queue, another MQGET */
/* must be issued to retrieve the message. If other */
/* MQM calls have been made in the intervening */
/* period, this may necessitate reinitializing the */
/* MQMD and MQGMO structures. */
/* In this code, no intervening calls */
/* have been made, so the only change required to */
/* the structures is to specify MQGMO_NO_WAIT, */
/* since we now know the message is there. */
/* This code uses the EXEC CICS DELAY command to */
/* suspend the program for a second. A batch program */
/* may achieve the same effect by calling an */
/* assembler language subroutine which issues a */
/* z/OS STIMER macro.
/*-----*/

```

```

if (signal_sw == 1)
{
    endloop = 0;
    do
    {
        EXEC CICS DELAY FOR HOURS(0) MINUTES(0) SECONDS(1);
        work_ecb = q_ecb & mask;
        switch (work_ecb)
        {
            case (MQEC_MSG_ARRIVED):

```



```

        endloop = 1;
        mqgmo_options = MQGMO_NO_WAIT;
        MQGET(Hconn, Hobj, &MsgDesc, &GetMsgOpts,
            BufferLength, message_buffer,
            &DataLength, &CompCode, &Reason);
        if (CompCode != MQCC_OK)
            ; /* Perform error processing. */
        break;
        case (MQEC_WAIT_INTERVAL_EXPIRED):
        case (MQEC_WAIT_CANCELED):
            endloop = 1;
            break;
        default:
            break;
    }
} while (endloop == 0);
}
return;
}

```

### **Interrogation des attributs d'un objet**

Cet exemple montre comment utiliser l'appel MQINQ pour demander les attributs d'une file d'attente.

Cette extraction est extraite de l'exemple d'application Attributs de file d'attente (programme CSQ4CCC1) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```

#include <cmqc.h> /* MQ API header file */
:
#define NUMBEROFSELECTORS 2

const MQHCONN Hconn = MQHC_DEF_HCONN;
:
static void InquireGetAndPut(char *Message,
                            PMQHOBJ pHobj,
                            char *Object)
{
    /* Declare local variables */
    /* SelectorCount = NUMBEROFSELECTORS;
       /* Number of selectors */
    /* IntAttrCount = NUMBEROFSELECTORS;
       /* Number of int attrs */
    /* CharAttrLength = 0;
       /* Length of char attribute buffer */
    MQCHAR *CharAttrs ;
    /* Character attribute buffer */
    /* SelectorTable[NUMBEROFSELECTORS];
       /* attribute selectors */
    /* IntAttrsTable[NUMBEROFSELECTORS];
       /* integer attributes */
    /* CompCode;
       /* Completion code */
    /* Reason;
       /* Qualifying reason */
    /* Open the queue. If successful, do the inquire
    /* call.
    /* Initialize the variables for the inquire
    /* call:
    /* - Set SelectorTable to the attributes whose
    /* status is
    /* required
    /* - All other variables are already set
    /* SelectorTable[0] = MQIA_INHIBIT_GET;
    SelectorTable[1] = MQIA_INHIBIT_PUT;
    /* Issue the inquire call
    /* Test the output of the inquire call. If the
    /* call failed, display an error message
    /* showing the completion code and reason code,
    /* otherwise display the status of the
    /* INHIBIT-GET and INHIBIT-PUT attributes
}

```

```

/*                                          */
MQINQ(Hconn,
      *pHobj,
      SelectorCount,
      SelectorsTable,
      IntAttrCount,
      IntAttrsTable,
      CharAttrLength,
      CharAttrs,
      &CompCode,
      &Reason);
if (CompCode != MQCC_OK)
{
    sprintf(Message, MESSAGE_4_E,
            ERROR_IN_MQINQ, CompCode, Reason);
    SetMsg(Message);
}
else
{
    /* Process the changes */
} /* end if CompCode */

```

### **Définition des attributs d'une file d'attente**

Cet exemple montre comment utiliser l'appel MQSET pour modifier les attributs d'une file d'attente.

Cette extraction est extraite de l'exemple d'application Attributs de file d'attente (programme CSQ4CCC1) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```

#include <cmqc.h>      /* MQ API header file      */
:
#define NUMBEROFSELECTORS 2

const MQHCONN Hconn = MQHC_DEF_HCONN;

static void InhibitGetAndPut(char *Message,
                             PMQHOBJ pHobj,
                             char *Object)
{
    /*                                          */
    /*      Declare local variables          */
    /*                                          */
    MQLONG SelectorCount = NUMBEROFSELECTORS;
    /*                                          */
    /*      Number of selectors              */
    MQLONG IntAttrCount = NUMBEROFSELECTORS;
    /*                                          */
    /*      Number of int attrs              */
    MQLONG CharAttrLength = 0;
    /*                                          */
    /*      Length of char attribute buffer   */
    MQCHAR *CharAttrs ;
    /*                                          */
    /*      Character attribute buffer        */
    MQLONG SelectorsTable[NUMBEROFSELECTORS];
    /*                                          */
    /*      attribute selectors              */
    MQLONG IntAttrsTable[NUMBEROFSELECTORS];
    /*                                          */
    /*      integer attributes              */
    MQLONG CompCode;
    /*                                          */
    /*      Completion code                  */
    MQLONG Reason;
    /*                                          */
    /*      Qualifying reason                */
    :
    /*                                          */
    /*      Open the queue. If successful, do the */
    /*      inquire call.                       */
    /*                                          */
    :
    /*                                          */
    /*      Initialize the variables for the set call: */
    /*      - Set SelectorsTable to the attributes to be */
    /*      set */
    /*      - Set IntAttrsTable to the required status */
    /*      - All other variables are already set */
    /*                                          */
    SelectorsTable[0] = MQIA_INHIBIT_GET;
    SelectorsTable[1] = MQIA_INHIBIT_PUT;
    IntAttrsTable[0] = MQQA_GET_INHIBITED;
    IntAttrsTable[1] = MQQA_PUT_INHIBITED;
    :
}

```

```

/* */
/* Issue the set call. */
/* Test the output of the set call. If the */
/* call fails, display an error message */
/* showing the completion code and reason */
/* code; otherwise move INHIBITED to the */
/* relevant screen map fields */
/* */
MQSET(Hconn,
      *pHobj,
      SelectorCount,
      SelectorsTable,
      IntAttrCount,
      IntAttrsTable,
      CharAttrLength,
      CharAttrs,
      &CompCode,
      &Reason);
if (CompCode != MQCC_OK)
{
    sprintf(Message, MESSAGE_4_E,
            ERROR_IN_MQSET, CompCode, Reason);
    SetMsg(Message);
}
else
{
    /* Process the changes */
} /* end if CompCode */

```

### **Extraction des informations de statut à l'aide de MQSTAT**

Cet exemple montre comment émettre une commande MQPUT asynchrone et extraire les informations de statut avec MQSTAT.

Cette extraction est extraite de l'exemple d'application Appeler MQSTAT (programme amqsapt0) fourni avec les systèmes IBM MQ for Windows . Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```

/*****/
/* */
/* Program name: AMQSAPT0 */
/* */
/* Description: Sample C program that asynchronously puts messages */
/* to a message queue (example using MQPUT & MQSTAT). */
/* */
/* Licensed Materials - Property of IBM */
/* */
/* 63H9336 */
/* (c) Copyright IBM Corp. 2006, 2024. All Rights Reserved. */
/* */
/* US Government Users Restricted Rights - Use, duplication or */
/* disclosure restricted by GSA ADP Schedule Contract with */
/* IBM Corp. */
/* */
/*****/
/* */
/* Function: */
/* */
/* AMQSAPT0 is a sample C program to put messages on a message */
/* queue with asynchronous response option, querying the success */
/* of the put operations with MQSTAT. */
/* */
/* -- messages are sent to the queue named by the parameter */
/* */
/* -- gets lines from StdIn, and adds each to target */
/* queue, taking each line of text as the content */
/* of a datagram message; the sample stops when a null */
/* line (or EOF) is read. */
/* New-line characters are removed. */
/* If a line is longer than 99 characters it is broken up */
/* into 99-character pieces. Each piece becomes the */
/* content of a datagram message. */
/* If the length of a line is a multiple of 99 plus 1, for */
/* example, 199, the last piece will only contain a */
/* new-line character so will terminate the input. */
/* */

```

```

/*
/* -- writes a message for each MQI reason other than
/* MQRC_NONE; stops if there is a MQI completion code
/* of MQCC_FAILED
/*
/* -- summarizes the overall success of the put operations
/* through a call to MQSTAT to query MQSTAT_TYPE_ASYNC_ERROR*/
/*
/* Program logic:
/* MQOPEN target queue for OUTPUT
/* while end of input file not reached,
/* . read next line of text
/* . MQPUT datagram message with text line as data
/* MQCLOSE target queue
/* MQSTAT connection
/*
/*
/*
/*****
/*
/* AMQSAPT0 has the following parameters
/* required:
/* (1) The name of the target queue
/* optional:
/* (2) Queue manager name
/* (3) The open options
/* (4) The close options
/* (5) The name of the target queue manager
/* (6) The name of the dynamic queue
/*
/*****
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
/* includes for MQI */
#include <cmqc.h>

int main(int argc, char **argv)
{
/* Declare file and character for sample input
FILE *fp;

/* Declare MQI structures needed
MQOD od = {MQOD_DEFAULT}; /* Object Descriptor
MQMD md = {MQMD_DEFAULT}; /* Message Descriptor
MQPMO pmo = {MQPMO_DEFAULT}; /* put message options
MQSTS sts = {MQSTS_DEFAULT}; /* status information
/** note, sample uses defaults where it can **/
MQHCONN Hcon; /* connection handle
MQHOBJ Hobj; /* object handle
MQLONG O_options; /* MQOPEN options
MQLONG C_options; /* MQCLOSE options
MQLONG CompCode; /* completion code
MQLONG OpenCode; /* MQOPEN completion code
MQLONG Reason; /* reason code
MQLONG CReason; /* reason code for MQCONN
MQLONG messlen; /* message length
char buffer[100]; /* message buffer
char QMName[50]; /* queue manager name

printf("Sample AMQSAPT0 start\n");
if (argc < 2)
{
printf("Required parameter missing - queue name\n");
exit(99);
}

/*****
/*
/* Connect to queue manager
/*
/*****
QMName[0] = 0; /* default */
if (argc > 2)
strcpy(QMName, argv[2]);
MQCONN(QMName, /* queue manager
&Hcon, /* connection handle
&Compcode, /* completion code
&Reason); /* reason code
/* report reason and stop if it failed */
if (CompCode == MQCC_FAILED)
{
printf("MQCONN ended with reason code %d\n", CReason);

```

```

    exit( (int)CReason );
}

/*****
/*
/*   Use parameter as the name of the target queue
/*
/*
*****/
strncpy(od.ObjectName, argv[1], (size_t)MQ_Q_NAME_LENGTH);
printf("target queue is %s\n", od.ObjectName);

if (argc > 5)
{
    strncpy(od.ObjectQMgrName, argv[5], (size_t) MQ_Q_MGR_NAME_LENGTH);
    printf("target queue manager is %s\n", od.ObjectQMgrName);
}

if (argc > 6)
{
    strncpy(od.DynamicQName, argv[6], (size_t) MQ_Q_NAME_LENGTH);
    printf("dynamic queue name is %s\n", od.DynamicQName);
}

/*****
/*
/*   Open the target message queue for output
/*
/*
*****/
if (argc > 3)
{
    O_options = atoi( argv[3] );
    printf("open options are %d\n", O_options);
}
else
{
    O_options = MQOO_OUTPUT          /* open queue for output      */
                | MQOO_FAIL_IF_QUIESCING /* but not if MQM stopping */
                ;                  /* = 0x2010 = 8208 decimal */
}

MQOPEN(Hcon,          /* connection handle      */
        &od,          /* object descriptor for queue */
        O_options,    /* open options           */
        &Hobj,        /* object handle          */
        &OpenCode,    /* MQOPEN completion code */
        &Reason);    /* reason code            */

/* report reason, if any; stop if failed */
if (Reason != MQRC_NONE)
{
    printf("MQOPEN ended with reason code %d\n", Reason);
}

if (OpenCode == MQCC_FAILED)
{
    printf("unable to open queue for output\n");
}

/*****
/*
/*   Read lines from the file and put them to the message queue
/*   Loop until null line or end of file, or there is a failure
/*
*****/
CompCode = OpenCode; /* use MQOPEN result for initial test */
fp = stdin;

memcpy(md.Format, /* character string format */
        MQFMT_STRING, (size_t)MQ_FORMAT_LENGTH);

/*****
/* These options specify that put operation should occur
/* asynchronously and the application will check the success
/* using MQSTAT at a later time.
*****/
md.Persistence = MQPER_NOT_PERSISTENT;
pmo.Options |= MQPMO_ASYNC_RESPONSE;

/*****
/* These options cause the MsgId and CorrelId to be replaced, so
/* that there is no need to reset them before each MQPUT
*****/

```

```

pmo.Options |= MQPMO_NEW_MSG_ID;
pmo.Options |= MQPMO_NEW_CORREL_ID;

while (CompCode != MQCC_FAILED)
{
    if (fgets(buffer, sizeof(buffer), fp) != NULL)
    {
        messlen = (MQLONG)strlen(buffer); /* length without null */
        if (buffer[messlen-1] == '\n') /* last char is a new-line */
        {
            buffer[messlen-1] = '\0'; /* replace new-line with null */
            --messlen; /* reduce buffer length */
        }
    }
    else messlen = 0; /* treat EOF same as null line */

    /******
    /* Put each buffer to the message queue */
    /******
    if (messlen > 0)
    {
        MQPUT(Hcon, /* connection handle */
            Hobj, /* object handle */
            &md, /* message descriptor */
            &pmo, /* default options (datagram) */
            messlen, /* message length */
            buffer, /* message buffer */
            &CompCode, /* completion code */
            &Reason); /* reason code */

        /* report reason, if any */
        if (Reason != MQRC_NONE)
        {
            printf("MQPUT ended with reason code %d\n", Reason);
        }
    }
    else /* satisfy end condition when empty line is read */
        CompCode = MQCC_FAILED;
}

/******
/* Close the target queue (if it was opened) */
/******
if (OpenCode != MQCC_FAILED)
{
    if (argc > 4)
    {
        C_options = atoi( argv[4] );
        printf("close options are %d\n", C_options);
    }
    else
    {
        C_options = MQCO_NONE; /* no close options */
    }

    MQCLOSE(Hcon, /* connection handle */
        &Hobj, /* object handle */
        C_options, /* completion code */
        &CompCode, /* reason code */
        &Reason);

    /* report reason, if any */
    if (Reason != MQRC_NONE)
    {
        printf("MQCLOSE ended with reason code %d\n", Reason);
    }
}

/******
/* Query how many asynchronous puts succeeded */
/******
MQSTAT(&Hcon, /* connection handle */
    MQSTAT_TYPE_ASYNC_ERROR, /* status type */
    &Sts, /* MQSTS structure */
    &CompCode, /* completion code */
    &Reason); /* reason code */

```

```

/* report reason, if any      */
if (Reason != MQRC_NONE)
{
    printf("MQSTAT ended with reason code %d\n", Reason);
}
else
{
    /* Display results */
    printf("Succeeded putting %d messages\n",
           sts.PutSuccessCount);
    printf("%d messages were put with a warning\n",
           sts.PutWarningCount);
    printf("Failed to put %d messages\n",
           sts.PutFailureCount);

    if(sts.CompCode == MQCC_WARNING)
    {
        printf("The first warning that occurred had reason code %d\n",
               sts.Reason);
    }
    else if(sts.CompCode == MQCC_FAILED)
    {
        printf("The first error that occurred had reason code %d\n",
               sts.Reason);
    }
}

/*****
/*
/*  Disconnect from MQM if not already connected      */
/*
/*
/*****
if (CReason != MQRC_ALREADY_CONNECTED)
{
    MQDISC(&Hcon,           /* connection handle      */
           &CompCode,      /* completion code       */
           &Reason);      /* reason code           */

    /* report reason, if any      */
    if (Reason != MQRC_NONE)
    {
        printf("MQDISC ended with reason code %d\n", Reason);
    }
}

/*****
/*
/*  END OF AMQSAPT0      */
/*
/*****
printf("Sample AMQSAPT0 end\n");
return(0);
}

```

## Exemples COBOL

Cette collection de rubriques est extraite des exemples d'application IBM MQ for z/OS . Elles sont applicables à toutes les plateformes, sauf indication contraire.

### **Connexion à un gestionnaire de files d'attente**

Cet exemple montre comment utiliser l'appel MQCONN pour connecter un programme à un gestionnaire de files d'attente dans z/OS batch.

Cette extraction est extraite de l'exemple d'application Parcourir (programme CSQ4BVA1) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```

* -----*
WORKING-STORAGE SECTION.
* -----*
*   W02 - Data fields derived from the PARM field
01  W02-MQM          PIC X(48) VALUE SPACES.
*   W03 - MQM API fields
01  W03-HCONN       PIC S9(9) BINARY.

```

```

01 W03-COMPCODE          PIC S9(9) BINARY.
01 W03-REASON           PIC S9(9) BINARY.
*
*   MQV contains constants (for filling in the control
*   blocks)
*   and return codes (for testing the result of a call)
*
01 W05-MQM-CONSTANTS.
COPY CMQV SUPPRESS.
:
*   Separate into the relevant fields any data passed
*   in the PARM statement
*
UNSTRING PARM-STRING DELIMITED BY ALL ','
                INTO W02-MQM
                W02-OBJECT.
:
*   Connect to the specified queue manager.
*
CALL 'MQCONN' USING W02-MQM
                  W03-HCONN
                  W03-COMPCODE
                  W03-REASON.
*
*   Test the output of the connect call.  If the call
*   fails, print an error message showing the
*   completion code and reason code.
*
IF (W03-COMPCODE NOT = MQCC-OK) THEN
:
END-IF.
:

```

### ***Déconnexion d'un gestionnaire de files d'attente***

Cet exemple montre comment utiliser l'appel MQDISC pour déconnecter un programme d'un gestionnaire de files d'attente dans un lot z/OS .

Les variables utilisées dans cette extraction de code sont celles qui ont été définies dans «Connexion à un gestionnaire de files d'attente», à la page 23. Cette extraction est extraite de l'exemple d'application Parcourir (programme CSQ4BVA1) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```

:
*
*   Disconnect from the queue manager
*
CALL 'MQDISC' USING W03-HCONN
                  W03-COMPCODE
                  W03-REASON.
*
*   Test the output of the disconnect call.  If the
*   call fails, print an error message showing the
*   completion code and reason code.
*
IF (W03-COMPCODE NOT = MQCC-OK) THEN
:
END-IF.
:

```

### ***Création d'une file d'attente dynamique***

Cet exemple montre comment utiliser l'appel MQOPEN pour créer une file d'attente dynamique.

Cet extrait provient de l'exemple d'application Credit Check (programme CSQ4CVB1) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```

:
* -----*
WORKING-STORAGE SECTION.
* -----*
*

```



```

*   W02 - Queues processed in this program
*
01  W02-MODEL-QNAME      PIC X(48) VALUE
    'CSQ4SAMP.B1.MODEL          '
01  W02-NAME-PREFIX     PIC X(48) VALUE
    'CSQ4SAMP.B1.*             '
01  W02-TEMPORARY-Q     PIC X(48).
*
*   W03 - MQM API fields
*
01  W03-HCONN          PIC S9(9) BINARY VALUE ZERO.
01  W03-OPTIONS       PIC S9(9) BINARY.
01  W03-HOBJ          PIC S9(9) BINARY.
01  W03-COMPCODE      PIC S9(9) BINARY.
01  W03-REASON        PIC S9(9) BINARY.
*
*   API control blocks
*
01  MQM-OBJECT-DESCRIPTOR.
    COPY CMQODV.
*
*   CMQV contains constants (for setting or testing
*   field values) and return codes (for testing the
*   result of a call)
*
01  MQM-CONSTANTS.
    COPY CMQV SUPPRESS.
* -----*
PROCEDURE DIVISION.
* -----*
:
* -----*
OPEN-TEMP-RESPONSE-QUEUE SECTION.
* -----*

```

```

*
*   This section creates a temporary dynamic queue
*   using a model queue
*
* -----*
*
*   Change three fields in the Object Descriptor (MQOD)
*   control block. (MQODV initializes the other fields)
*
    MOVE MQOT-Q          TO MQOD-OBJECTTYPE.
    MOVE W02-MODEL-QNAME TO MQOD-OBJECTNAME.
    MOVE W02-NAME-PREFIX TO MQOD-DYNAMICQNAME.
*
    COMPUTE W03-OPTIONS = MQOD-INPUT-EXCLUSIVE.
*
    CALL 'MQOPEN' USING W03-HCONN
                      MQOD
                      W03-OPTIONS
                      W03-HOBJ-MODEL
                      W03-COMPCODE
                      W03-REASON.
*
    IF W03-COMPCODE NOT = MQCC-OK
        MOVE 'MQOPEN'      TO M01-MSG4-OPERATION
        MOVE W03-COMPCODE  TO M01-MSG4-COMPCODE
        MOVE W03-REASON    TO M01-MSG4-REASON
        MOVE M01-MESSAGE-4 TO M00-MESSAGE
    ELSE
        MOVE MQOD-OBJECTNAME TO W02-TEMPORARY-Q
    END-IF.
*
    OPEN-TEMP-RESPONSE-QUEUE-EXIT.
*
*   Return to performing section.
*
    EXIT.
    EJECT
*

```

### ***Ouverture d'une file d'attente existante***

Cet exemple montre comment utiliser l'appel MQOPEN pour ouvrir une file d'attente existante.

Cette extraction est extraite de l'exemple d'application Parcourir (programme CSQ4BVA1) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```

:
* -----*
WORKING-STORAGE SECTION.
* -----*
*
*   W01 - Fields derived from the command area input
*
01 W01-OBJECT          PIC X(48).
*
*   W02 - MQM API fields
*
01 W02-HCONN          PIC S9(9) BINARY VALUE ZERO.
01 W02-OPTIONS        PIC S9(9) BINARY.
01 W02-HOBJ           PIC S9(9) BINARY.
01 W02-COMPCODE       PIC S9(9) BINARY.
01 W02-REASON         PIC S9(9) BINARY.
*
*   CMQODV defines the object descriptor (MQOD)
*
01 MQM-OBJECT-DESCRIPTOR.
   COPY CMQODV.
*
*   CMQV contains constants (for setting or testing
*   field values) and return codes (for testing the
*   result of a call)
*
01 MQM-CONSTANTS.
   COPY CMQV SUPPRESS.
* -----*
E-OPEN-QUEUE SECTION.
* -----*
*
*   This section opens the queue
*
*   Initialize the Object Descriptor (MQOD) control
*   block
*   (The copy file initializes the remaining fields.)
*
MOVE MQOT-Q           TO MQOD-OBJECTTYPE.
MOVE W01-OBJECT       TO MQOD-OBJECTNAME.
*
*   Initialize W02-OPTIONS to open the queue for both
*   inquiring about and setting attributes
*
COMPUTE W02-OPTIONS = MQ00-INQUIRE + MQ00-SET.

```

```

*
*   Open the queue
*
CALL 'MQOPEN' USING W02-HCONN
                   MQOD
                   W02-OPTIONS
                   W02-HOBJ
                   W02-COMPCODE
                   W02-REASON.
*
*   Test the output from the open
*
*   If the completion code is not OK, display a
*   separate error message for each of the following
*   errors:
*
*   Q-MGR-NOT-AVAILABLE - MQM is not available
*   CONNECTION-BROKEN   - MQM is no longer connected to CICS
*   UNKNOWN-OBJECT-NAME - The queue does not exist
*   NOT-AUTHORIZED      - The user is not authorized to open
*                       the queue
*
*   For any other error, display an error message
*   showing the completion and reason codes
*
IF W02-COMPCODE NOT = MQCC-OK
EVALUATE TRUE

```

```

*
  WHEN W02-REASON = MQRC-Q-MGR-NOT-AVAILABLE
    MOVE M01-MESSAGE-6 TO M00-MESSAGE
*
  WHEN W02-REASON = MQRC-CONNECTION-BROKEN
    MOVE M01-MESSAGE-6 TO M00-MESSAGE
*
  WHEN W02-REASON = MQRC-UNKNOWN-OBJECT-NAME
    MOVE M01-MESSAGE-2 TO M00-MESSAGE
*
  WHEN W02-REASON = MQRC-NOT-AUTHORIZED
    MOVE M01-MESSAGE-3 TO M00-MESSAGE
*
  WHEN OTHER
    MOVE 'M0OPEN'      TO M01-MSG4-OPERATION
    MOVE W02-COMPCODE TO M01-MSG4-COMPCODE
    MOVE W02-REASON   TO M01-MSG4-REASON
    MOVE M01-MESSAGE-4 TO M00-MESSAGE
  END-EVALUATE
END-IF.
E-EXIT.
*
*   Return to performing section
*
  EXIT.
  EJECT

```

### ***Fermeture d'une file d'attente***

Cet exemple montre comment utiliser l'appel MQCLOSE.

Les variables utilisées dans cette extraction de code sont celles qui ont été définies dans [«Connexion à un gestionnaire de files d'attente»](#), à la page 23. Cette extraction est extraite de l'exemple d'application [Parcourir \(programme CSQ4BVA1\)](#) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```

:
*
*   Close the queue
*
  MOVE MQCO-NONE TO W03-OPTIONS.
*
  CALL 'MQCLOSE' USING W03-HCONN
                      W03-HOBJ
                      W03-OPTIONS
                      W03-COMPCODE
                      W03-REASON.
*
*   Test the output of the MQCLOSE call.  If the call
*   fails, print an error message showing the
*   completion code and reason code.
*
  IF (W03-COMPCODE NOT = MQCC-OK) THEN
    MOVE 'CLOSE'      TO W04-MSG4-TYPE
    MOVE W03-COMPCODE TO W04-MSG4-COMPCODE
    MOVE W03-REASON   TO W04-MSG4-REASON
    MOVE W04-MESSAGE-4 TO W00-PRINT-DATA
    PERFORM PRINT-LINE
    MOVE W06-CSQ4-ERROR TO W00-RETURN-CODE
  END-IF.
*

```

### ***Insertion d'un message à l'aide de MQPUT***

Cet exemple montre comment utiliser l'appel MQPUT à l'aide du contexte.

Cet extrait provient de l'exemple d'application [Credit Check \(programme CSQ4CVB1\)](#) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```

:
* -----*
  WORKING-STORAGE SECTION.

```

```

* -----*
*
*   W02 - Queues processed in this program
*
* 01 W02-TEMPORARY-Q          PIC X(48).
*
*   W03 - MQM API fields
*
* 01 W03-HCONN              PIC S9(9) BINARY VALUE ZERO.
* 01 W03-HOBJ-INQUIRY       PIC S9(9) BINARY.
* 01 W03-OPTIONS            PIC S9(9) BINARY.
* 01 W03-BUFFLEN            PIC S9(9) BINARY.
* 01 W03-COMPCODE           PIC S9(9) BINARY.
* 01 W03-REASON             PIC S9(9) BINARY.
*
* 01 W03-PUT-BUFFER.
*
*   05 W03-CSQ4BIIM.
*   COPY CSQ4VB1.
*
*   API control blocks
*
* 01 MQM-MESSAGE-DESCRIPTOR.
*   COPY CMQMDV.
* 01 MQM-PUT-MESSAGE-OPTIONS.
*   COPY CMQPMOV.
*
*   MQV contains constants (for filling in the
*   control blocks) and return codes (for testing
*   the result of a call).
*
* 01 MQM-CONSTANTS.
*   COPY CMQV SUPPRESS.
* -----*
*   PROCEDURE DIVISION.
* -----*
*
*   Open queue and build message.
*
*

```

```

*
* Set the message descriptor and put-message options to
* the values required to create the message.
* Set the length of the message.
*
MOVE MQMT-REQUEST          TO MQMD-MSGTYPE.
MOVE MQCI-NONE             TO MQMD-CORRELID.
MOVE MQMI-NONE             TO MQMD-MSGID.
MOVE W02-TEMPORARY-Q      TO MQMD-REPLYTOQ.
MOVE SPACES                TO MQMD-REPLYTOQMGR.
MOVE 5                     TO MQMD-PRIORITY.
MOVE MQPER-NOT-PERSISTENT TO MQMD-PERSISTENCE.
COMPUTE MQPMO-OPTIONS     = MQPMO-NO-SYNCPOINT +
                           MQPMO-DEFAULT-CONTEXT.
MOVE LENGTH OF CSQ4BIIM-MSG TO W03-BUFFLEN.
*
CALL 'MQPUT' USING W03-HCONN
                  W03-HOBJ-INQUIRY
                  MQMD
                  MQPMO
                  W03-BUFFLEN
                  W03-PUT-BUFFER
                  W03-COMPCODE
                  W03-REASON.
IF W03-COMPCODE NOT = MQCC-OK
:
:
END-IF.

```

### ***Insertion d'un message à l'aide de MQPUT1***

Cet exemple montre comment utiliser l'appel MQPUT1 .

Cet extrait est extrait de l'exemple d'application Credit Check (programme CSQ4CVB5) fourni avec IBM MQ pour z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```

:
* -----*
WORKING-STORAGE SECTION.
* -----*
*
*   W03 - MQM API fields
*
01 W03-HCONN          PIC S9(9) BINARY VALUE ZERO.
01 W03-OPTIONS       PIC S9(9) BINARY.
01 W03-COMPCODE      PIC S9(9) BINARY.
01 W03-REASON        PIC S9(9) BINARY.
01 W03-BUFFLEN       PIC S9(9) BINARY.
*
01 W03-PUT-BUFFER.
05 W03-CSQ4BQRM.
COPY CSQ4VB4.

*
*   API control blocks
*
01 MQM-OBJECT-DESCRIPTOR.
COPY CMQODV.
01 MQM-MESSAGE-DESCRIPTOR.
COPY CMQMDV.
01 MQM-PUT-MESSAGE-OPTIONS.
COPY CMQPMOV.
*
* CMQV contains constants (for filling in the
* control blocks) and return codes (for testing
* the result of a call).
*
01 MQM-MQV.
COPY CMQV SUPPRESS.
* -----*
PROCEDURE DIVISION.
* -----*
:
*   Get the request message.
:
* -----*
PROCESS-QUERY SECTION.
* -----*
:
*   Build the reply message.
:
*
* Set the object descriptor, message descriptor and
* put-message options to the values required to create
* the message.
* Set the length of the message.
*
MOVE MQMD-REPLYTOQ    TO MQOD-OBJECTNAME.
MOVE MQMD-REPLYTOQMGR TO MQOD-OBJECTQMGRNAME.
MOVE MQMT-REPLY      TO MQMD-MSGTYPE.
MOVE SPACES          TO MQMD-REPLYTOQ.
MOVE SPACES          TO MQMD-REPLYTOQMGR.
MOVE LOW-VALUES      TO MQMD-MSGID.
COMPUTE MQPMO-OPTIONS = MQPMO-SYNCPPOINT +
                        MQPMO-PASS-IDENTITY-CONTEXT.
MOVE W03-HOBJ-CHECKQ TO MQPMO-CONTEXT.
MOVE LENGTH OF CSQ4BQRM-MSG TO W03-BUFFLEN.
*
CALL 'MQPUT1' USING W03-HCONN
                   MQOD
                   MQMD
                   MQPMO
                   W03-BUFFLEN
                   W03-PUT-BUFFER
                   W03-COMPCODE
                   W03-REASON.
IF W03-COMPCODE NOT = MQCC-OK
  MOVE 'MQPUT1'      TO M02-OPERATION
  MOVE MQOD-OBJECTNAME TO M02-OBJECTNAME

```

```

PERFORM RECORD-CALL-ERROR
PERFORM FORWARD-MSG-TO-DLQ
END-IF.

```

\*

### **obtention d'un message**

Cet exemple montre comment utiliser l'appel MQGET pour supprimer un message d'une file d'attente.

Cet extrait provient de l'exemple d'application Credit Check (programme CSQ4CVB1) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```

:
* -----*
WORKING-STORAGE SECTION.
* -----*
*
*   W03 - MQM API fields
*
01 W03-HCONN          PIC S9(9) BINARY VALUE ZERO.
01 W03-HOBJ-RESPONSE PIC S9(9) BINARY.
01 W03-OPTIONS       PIC S9(9) BINARY.
01 W03-BUFFLEN       PIC S9(9) BINARY.
01 W03-DATALEN       PIC S9(9) BINARY.
01 W03-COMPCODE      PIC S9(9) BINARY.
01 W03-REASON        PIC S9(9) BINARY.
*
01 W03-GET-BUFFER.
   05 W03-CSQ4BAM.
   COPY CSQ4VB2.
*
*   API control blocks
*
01 MQM-MESSAGE-DESCRIPTOR.
   COPY CMQMDV.
01 MQM-GET-MESSAGE-OPTIONS.
   COPY CMQGMV.
*
*   MQV contains constants (for filling in the
*   control blocks) and return codes (for testing
*   the result of a call).
*
01 MQM-CONSTANTS.
   COPY CMQV SUPPRESS.
* -----*
A-MAIN SECTION.
* -----*
:
*   Open response queue.
:
* -----*
PROCESS-RESPONSE-SCREEN SECTION.
* -----*
*
*   This section gets a message from the response queue.
*
*   When a correct response is received, it is
*   transferred to the map for display; otherwise
*   an error message is built.
* -----*

```

```

*
*   Set get-message options
*
COMPUTE MQGMO-OPTIONS = MQGMO-SYNCPPOINT +
                       MQGMO-ACCEPT-TRUNCATED-MSG +
                       MQGMO-NO-WAIT.
*
*   Set msgid and correlid in MQMD to nulls so that any
*   message will qualify.
*   Set length to available buffer length.
*
MOVE MQMI-NONE TO MQMD-MSGID.
MOVE MQCI-NONE TO MQMD-CORRELID.

```

```

MOVE LENGTH OF W03-GET-BUFFER TO W03-BUFFLEN.
*
CALL 'MQGET' USING W03-HCONN
                  W03-HOBJ-RESPONSE
                  MQMD
                  MQGMO
                  W03-BUFFLEN
                  W03-GET-BUFFER
                  W03-DATALEN
                  W03-COMPCODE
                  W03-REASON.

EVALUATE TRUE
  WHEN W03-COMPCODE NOT = MQCC-FAILED
  :
  :
  *      Process the message
  :
  WHEN (W03-COMPCODE = MQCC-FAILED AND
        W03-REASON = MQRC-NO-MSG-AVAILABLE)
    MOVE M01-MESSAGE-9 TO M00-MESSAGE
    PERFORM CLEAR-RESPONSE-SCREEN

  *
  WHEN OTHER
    MOVE 'MQGET '      TO M01-MSG4-OPERATION
    MOVE W03-COMPCODE TO M01-MSG4-COMPCODE
    MOVE W03-REASON   TO M01-MSG4-REASON
    MOVE M01-MESSAGE-4 TO M00-MESSAGE
    PERFORM CLEAR-RESPONSE-SCREEN

END-EVALUATE.

```

### ***Obtention d'un message à l'aide de l'option d'attente***

Cet exemple montre comment utiliser l'appel MQGET avec l'option d'attente et accepter les messages tronqués.

Cet extrait est extrait de l'exemple d'application Credit Check (programme CSQ4CVB5) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```

:
* -----*
WORKING-STORAGE SECTION.
* -----*
*
*   W00 - General work fields
*
01 W00-WAIT-INTERVAL   PIC S9(09) BINARY VALUE 30000.
*
*   W03 - MQM API fields
*
01 W03-HCONN          PIC S9(9) BINARY VALUE ZERO.
01 W03-OPTIONS        PIC S9(9) BINARY.
01 W03-HOBJ-CHECKQ    PIC S9(9) BINARY.
01 W03-COMPCODE       PIC S9(9) BINARY.
01 W03-REASON         PIC S9(9) BINARY.
01 W03-DATALEN        PIC S9(9) BINARY.
01 W03-BUFFLEN        PIC S9(9) BINARY.
*
01 W03-MSG-BUFFER.
   05 W03-CSQ4BCAQ.
   COPY CSQ4VB3.
*
*   API control blocks
*
01 MQM-MESSAGE-DESCRIPTOR.
   COPY CMQMDV.
01 MQM-GET-MESSAGE-OPTIONS.
   COPY CMQGMV.
*
*   CMQV contains constants (for filling in the
*   control blocks) and return codes (for testing
*   the result of a call).
*
01 MQM-MQV.
   COPY CMQV SUPPRESS.
* -----*
PROCEDURE DIVISION.
* -----*
:

```

```

*   Open input queue.
:

*
*   Get and process messages.
*
  COMPUTE MQGMO-OPTIONS = MQGMO-WAIT +
                        MQGMO-ACCEPT-TRUNCATED-MSG +
                        MQGMO-SYNCPOINT.
  MOVE LENGTH OF W03-MSG-BUFFER TO W03-BUFFLEN.
  MOVE W00-WAIT-INTERVAL TO MQGMO-WAITINTERVAL.
  MOVE MQMI-NONE TO MQMD-MSGID.
  MOVE MQCI-NONE TO MQMD-CORRELID.
*
*   Make the first MQGET call outside the loop.
*
  CALL 'MQGET' USING W03-HCONN
                    W03-HOBJ-CHECKQ
                    MQMD
                    MQGMO
                    W03-BUFFLEN
                    W03-MSG-BUFFER
                    W03-DATALEN
                    W03-COMPCODE
                    W03-REASON.

*
*   Test the output of the MQGET call using the
*   PERFORM loop that follows.
*
*   Perform whilst no failure occurs
*   - process this message
*   - reset the call parameters
*   - get another message
*   End-perform
*

*
*   Test the output of the MQGET call.  If the call
*   fails, send an error message showing the
*   completion code and reason code, unless the
*   completion code is NO-MSG-AVAILABLE.
*
  IF (W03-COMPCODE NOT = MQCC-FAILED) OR
     (W03-REASON NOT = MQRC-NO-MSG-AVAILABLE)
     MOVE 'MQGET '          TO M02-OPERATION
     MOVE MQ0D-OBJECTNAME  TO M02-OBJECTNAME
     PERFORM RECORD-CALL-ERROR
  END-IF.
:

```

### ***Obtention d'un message à l'aide de la notification***

Cet exemple montre comment utiliser l'appel MQGET avec la notification. Cet extrait provient de l'exemple d'application Credit Check (programme CSQ4CVB2) fourni avec IBM MQ for z/OS.

*La signalisation est disponible uniquement avec IBM MQ for z/OS .*

```

:
* -----*
WORKING-STORAGE SECTION.
* -----*
*
*   W00 - General work fields
*
01  W00-WAIT-INTERVAL    PIC S9(09) BINARY VALUE 30000.
*
*   W03 - MQM API fields
*
01  W03-HCONN           PIC S9(9) BINARY VALUE ZERO.
01  W03-HOBJ-REPLYQ    PIC S9(9) BINARY.
01  W03-COMPCODE       PIC S9(9) BINARY.
01  W03-REASON         PIC S9(9) BINARY.
01  W03-DATALEN        PIC S9(9) BINARY.
01  W03-BUFFLEN        PIC S9(9) BINARY.
:

```



```

01 W03-GET-BUFFER.
   05 W03-CSQ4BQRM.
   COPY CSQ4VB4.
*
*   05 W03-CSQ4BIIM REDEFINES W03-CSQ4BQRM.
*   COPY CSQ4VB1.
*
*   05 W03-CSQ4BPGM REDEFINES W03-CSQ4BIIM.
*   COPY CSQ4VB5.
*   :
*   API control blocks
*
01 MQM-MESSAGE-DESCRIPTOR.
   COPY CMQMDV.
01 MQM-GET-MESSAGE-OPTIONS.
   COPY CMQGMV.
   :
*   MQV contains constants (for filling in the
*   control blocks) and return codes (for testing
*   the result of a call).
*
01 MQM-MQV.
   COPY CMQV SUPPRESS.
* -----*
LINKAGE SECTION.
* -----*
01 L01-ECB-ADDR-LIST.
   05 L01-ECB-ADDR1      POINTER.
   05 L01-ECB-ADDR2      POINTER.
*
*
01 L02-ECBS.
   05 L02-INQUIRY-ECB1    PIC S9(09) BINARY.
   05 L02-REPLY-ECB2     PIC S9(09) BINARY.
01 REDEFINES L02-ECBS.
   05                      PIC X(02).
   05 L02-INQUIRY-ECB1-CC PIC S9(04) BINARY.
   05                      PIC X(02).
   05 L02-REPLY-ECB2-CC  PIC S9(04) BINARY.
*
* -----*
PROCEDURE DIVISION.
* -----*
:
* Initialize variables, open queues, set signal on
* inquiry queue.
:
* -----*
PROCESS-SIGNAL-ACCEPTED SECTION.
* -----*
* This section gets a message with signal.  If a      *
* message is received, process it.  If the signal    *
* is set or is already set, the program goes into    *
* an operating system wait.                          *
* Otherwise an error is reported and call error set. *
* -----*
*
PERFORM REPLYQ-GETSIGNAL.
*
EVALUATE TRUE
  WHEN (W03-COMPCODE = MQCC-OK AND
        W03-REASON = MQRC-NONE)
    PERFORM PROCESS-REPLYQ-MESSAGE
*
  WHEN (W03-COMPCODE = MQCC-WARNING AND
        W03-REASON = MQRC-SIGNAL-REQUEST-ACCEPTED)
    OR
    (W03-COMPCODE = MQCC-FAILED AND
     W03-REASON = MQRC-SIGNAL-OUTSTANDING)
    PERFORM EXTERNAL-WAIT
*
  WHEN OTHER
    MOVE 'MQGET SIGNAL' TO M02-OPERATION
    MOVE MQ0D-OBJECTNAME TO M02-OBJECTNAME
    PERFORM RECORD-CALL-ERROR
    MOVE W06-CALL-ERROR TO W06-CALL-STATUS
END-EVALUATE.
*
PROCESS-SIGNAL-ACCEPTED-EXIT.
*   Return to performing section

```

```

EXIT.
EJECT
*

* -----*
EXTERNAL-WAIT SECTION.
* -----*
* This section performs an external CICS wait on two      *
* ECBs until at least one is posted.  It then calls      *
* the sections to handle the posted ECB.                 *
* -----*
EXEC CICS WAIT EXTERNAL
      ECBLIST(W04-ECB-ADDR-LIST-PTR)
      NUMEVENTS(2)
END-EXEC.
*
* At least one ECB must have been posted to get to this
* point.  Test which ECB has been posted and perform
* the appropriate section.
*
IF L02-INQUIRY-ECB1 NOT = 0
  PERFORM TEST-INQUIRYQ-ECB
ELSE
  PERFORM TEST-REPLYQ-ECB
END-IF.
*
EXTERNAL-WAIT-EXIT.
*
Return to performing section.
*
EXIT.
EJECT
:
* -----*
REPLYQ-GETSIGNAL SECTION.
* -----*
* This section performs an MQGET call (in syncpoint with *
* signal) on the reply queue.  The signal field in the  *
* MQGMO is set to the address of the ECB.               *
* Response handling is done by the performing section.  *
* -----*
*
COMPUTE MQGMO-OPTIONS          = MQGMO-SYNCPPOINT +
                                MQGMO-SET-SIGNAL.
MOVE W00-WAIT-INTERVAL         TO MQGMO-WAITINTERVAL.
MOVE LENGTH OF W03-GET-BUFFER TO W03-BUFFLEN.
*
MOVE ZEROS                     TO L02-REPLY-ECB2.
SET MQGMO-SIGNAL1 TO ADDRESS OF L02-REPLY-ECB2.
*

*
* Set msgid and correlid to nulls so that any message
* will qualify.
*
MOVE MQMI-NONE TO MQMD-MSGID.
MOVE MQCI-NONE TO MQMD-CORRELID.
*
CALL 'MQGET' USING W03-HCONN
                  W03-HOBJ-REPLYQ
                  MQMD
                  MQGMO
                  W03-BUFFLEN
                  W03-GET-BUFFER
                  W03-DATALEN
                  W03-COMPCODE
                  W03-REASON.
*
REPLYQ-GETSIGNAL-EXIT.
*
Return to performing section.
*
EXIT.
EJECT
*
:

```

## Interrogation des attributs d'un objet

Cet exemple montre comment utiliser l'appel MQINQ pour demander les attributs d'une file d'attente.

Cette extraction est extraite de l'exemple d'application Attributs de file d'attente (programme CSQ4CVC1) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#).

```
:
* -----*
* WORKING-STORAGE SECTION.
* -----*
*
*   W02 - MQM API fields
*
01 W02-SELECTORCOUNT    PIC S9(9) BINARY VALUE 2.
01 W02-INTATTRCOUNT    PIC S9(9) BINARY VALUE 2.
01 W02-CHARATTRLENGTH   PIC S9(9) BINARY VALUE ZERO.
01 W02-CHARATTRS        PIC X      VALUE LOW-VALUES.
01 W02-HCONN            PIC S9(9) BINARY VALUE ZERO.
01 W02-HOBJ             PIC S9(9) BINARY.
01 W02-COMPCODE         PIC S9(9) BINARY.
01 W02-REASON           PIC S9(9) BINARY.
01 W02-SELECTORS-TABLE.
   05 W02-SELECTORS      PIC S9(9) BINARY OCCURS 2 TIMES
01 W02-INTATTRS-TABLE.
   05 W02-INTATTRS      PIC S9(9) BINARY OCCURS 2 TIMES
*
*   CMQODV defines the object descriptor (MQOD).
*
01 MQM-OBJECT-DESCRIPTOR.
   COPY CMQODV.
*
*   CMQV contains constants (for setting or testing field
*   values) and return codes (for testing the result of a
*   call).
*
01 MQM-CONSTANTS.
   COPY CMQV SUPPRESS.
* -----*
* PROCEDURE DIVISION.
* -----*
*
*   Get the queue name and open the queue.
*
*   :
*
*   Initialize the variables for the inquiry call:
*   - Set W02-SELECTORS-TABLE to the attributes whose
*   status is required
*   - All other variables are already set
*
MOVE MQIA-INHIBIT-GET TO W02-SELECTORS(1).
MOVE MQIA-INHIBIT-PUT TO W02-SELECTORS(2).

*
*   Inquire about the attributes.
*
CALL 'MQINQ' USING W02-HCONN,
                  W02-HOBJ,
                  W02-SELECTORCOUNT,
                  W02-SELECTORS-TABLE,
                  W02-INTATTRCOUNT,
                  W02-INTATTRS-TABLE,
                  W02-CHARATTRLENGTH,
                  W02-CHARATTRS,
                  W02-COMPCODE,
                  W02-REASON.
*
*   Test the output from the inquiry:
*
*   - If the completion code is not OK, display an error
*   message showing the completion and reason codes
*
*   - Otherwise, move the correct attribute status into
*   the relevant screen map fields
*
```

```

IF W02-COMPCODE NOT = MQCC-OK
  MOVE 'MQINQ' TO M01-MSG4-OPERATION
  MOVE W02-COMPCODE TO M01-MSG4-COMPCODE
  MOVE W02-REASON TO M01-MSG4-REASON
  MOVE M01-MESSAGE-4 TO M00-MESSAGE
*
ELSE
*   Process the changes.
  :
  :   END-IF.
  :

```

### Définition des attributs d'une file d'attente

Cet exemple montre comment utiliser l'appel MQSET pour modifier les attributs d'une file d'attente.

Cette extraction est extraite de l'exemple d'application Attributs de file d'attente (programme CSQ4CVC1) fourni avec IBM MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir [Exemples de programmes procéduraux \(plateformes sauf z/OS\)](#)

```

:
* -----*
WORKING-STORAGE SECTION.
* -----*
*
*   W02 - MQM API fields
*
01 W02-SELECTORCOUNT PIC S9(9) BINARY VALUE 2.
01 W02-INTATTRCOUNT PIC S9(9) BINARY VALUE 2.
01 W02-CHARATTRLENGTH PIC S9(9) BINARY VALUE ZERO.
01 W02-CHARATTRS PIC X VALUE LOW-VALUES.
01 W02-HCONN PIC S9(9) BINARY VALUE ZERO.
01 W02-HOBJ PIC S9(9) BINARY.
01 W02-COMPCODE PIC S9(9) BINARY.
01 W02-REASON PIC S9(9) BINARY.
01 W02-SELECTORS-TABLE.
05 W02-SELECTORS PIC S9(9) BINARY OCCURS 2 TIMES.
01 W02-INTATTRS-TABLE.
05 W02-INTATTRS PIC S9(9) BINARY OCCURS 2 TIMES.
*
*   CMQODV defines the object descriptor (MQOD).
*
01 MQM-OBJECT-DESCRIPTOR.
COPY CMQODV.
*
*   CMQV contains constants (for setting or testing
*   field values) and return codes (for testing the
*   result of a call).
*
01 MQM-CONSTANTS.
COPY CMQV SUPPRESS.
* -----*
PROCEDURE DIVISION.
* -----*

```

```

*
*   Get the queue name and open the queue.
*
*
*
*   Initialize the variables required for the set call:
* - Set W02-SELECTORS-TABLE to the attributes to be set
* - Set W02-INTATTRS-TABLE to the required status
* - All other variables are already set
*
MOVE MQIA-INHIBIT-GET TO W02-SELECTORS(1).
MOVE MQIA-INHIBIT-PUT TO W02-SELECTORS(2).
MOVE MQQA-GET-INHIBITED TO W02-INTATTRS(1).
MOVE MQQA-PUT-INHIBITED TO W02-INTATTRS(2).
*
*   Set the attributes.
*
CALL 'MQSET' USING W02-HCONN,
                  W02-HOBJ,
                  W02-SELECTORCOUNT,

```

```

W02-SELECTORS-TABLE,
W02-INTATTRCOUNT,
W02-INTATTRS-TABLE,
W02-CHARATTRLENGTH,
W02-CHARATTRS,
W02-COMPCODE,
W02-REASON.
*
* Test the output from the call:
*
* - If the completion code is not OK, display an error
*   message showing the completion and reason codes
*
* - Otherwise, move 'INHIBITED' into the relevant
*   screen map fields
*
IF W02-COMPCODE NOT = MQCC-OK
  MOVE 'MQSET'          TO M01-MSG4-OPERATION
  MOVE W02-COMPCODE    TO M01-MSG4-COMPCODE
  MOVE W02-REASON      TO M01-MSG4-REASON
  MOVE M01-MESSAGE-4  TO M00-MESSAGE
ELSE
*
*   Process the changes.
*
:
END-IF.

```

## Exemples de langage assembleur System/390

Cette collection de rubriques est principalement extraite des modèles d'application IBM MQ for z/OS .

### ***Connexion à un gestionnaire de files d'attente***

Cet exemple montre comment utiliser l'appel MQCONN pour connecter un programme à un gestionnaire de files d'attente dans z/OS batch.

Cette extraction est extraite de l'exemple de programme Browse (CSQ4BAA1) fourni avec IBM MQ for z/OS.

```

:
WORKAREA DSECT
*
PARMLIST CALL ,(0,0,0,0,0,0,0,0,0,0,0),VL,MF=L
*
COMPCODE DS    F           Completion code
REASON   DS    F           Reason code
HCONN   DS    F           Connection handle
      ORG
PARMADDR DS    F           Address of parm field
PARMLEN DS    H           Length of parm field
*
MQMNAME DS    CL48        Queue manager name
*
*
*****
* SECTION NAME : MAINPARM *
*****
MAINPARM DS    0H
      MVI  MQMNAME,X'40'
      MVC  MQMNAME+1(L'MQMNAME-1),MQMNAME
*
* Space out first byte and initialize
*
*
* Code to address and verify parameters passed omitted
*
*
PARM1MVE DS    0H
      SR   R1,R3           Length of data
      LA  R4,MQMNAME      Address for target
      BCTR R1,R0          Reduce for execute
      EX  R1,MOVEPARM     Move the data
*
*****
* EXECUTES *
*****

```

```
MOVEPARM MVC 0(*-*,R4),0(R3)
```

```
*  
EJECT
```

```
*****  
* SECTION NAME : MAINCONN *  
*****  
*  
*  
MAINCONN DS 0H  
XC HCONN,HCONN Null connection handle  
*  
CALL MQCONN, X  
(MQMNAME, X  
HCONN, X  
COMPCODE, X  
REASON), X  
MF=(E,PARMLIST),VL  
*  
LA R0,MQCC_OK Expected compcode  
C R0,COMPCODE As expected?  
BER R6 Yes .. return to caller  
*  
MVC INF4_TYP,=CL10'CONNECT '  
BAL R7,ERRCODE Translate error  
LA R0,8 Set exit code  
ST R0,EXITCODE to 8  
B ENDPROG End the program  
*
```

### Déconnexion d'un gestionnaire de files d'attente

Cet exemple montre comment utiliser l'appel MQDISC pour déconnecter un programme d'un gestionnaire de files d'attente dans un lot z/OS .

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```
:  
*  
* ISSUE MQI DISC REQUEST USING REENTRANT FORM  
* OF CALL MACRO  
*  
* HCONN WAS SET BY A PREVIOUS MQCONN REQUEST  
* R5 = WORK REGISTER  
*  
DISC DS 0H  
CALL MQDISC, X  
(HCONN, X  
COMPCODE, X  
REASON), X  
VL,MF=(E,CALLST)  
*  
LA R5,MQCC_OK  
C R5,COMPCODE  
BNE BADCALL  
:
```

```
BADCALL DS 0H  
:  
* CONSTANTS  
*  
CMQA  
*  
WORKING STORAGE (RE-ENTRANT)  
*  
WEG3 DSECT  
*  
CALLLST CALL ,(0,0,0,0,0,0,0,0,0,0,0,0),VL,MF=L  
*  
HCONN DS F  
COMPCODE DS F  
REASON DS F  
*  
*
```

```
LEG3    EQU *-WKEG3
        END
```

### ***Création d'une file d'attente dynamique***

Cet exemple montre comment utiliser l'appel MQOPEN pour créer une file d'attente dynamique.

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```

:
*
*   R5 = WORK REGISTER.
*
OPEN    DS    0H
*
        MVC   WOD_AREA,MQOD_AREA INITIALIZE WORKING VERSION OF
*                               MQOD WITH DEFAULTS
        MVC   WOD_OBJECTNAME,MOD_Q   COPY IN THE MODEL Q NAME
        MVC   WOD_DYNAMICQNAME,DYN_Q COPY IN THE DYNAMIC Q NAME
        L     R5,=AL4(MQOO_OUTPUT)   OPEN FOR OUTPUT AND
        A     R5,=AL4(MQOO_INQUIRE) INQUIRE
        ST    R5,OPTIONS

*
*   ISSUE MQI OPEN REQUEST USING REENTRANT
*   FORM OF CALL MACRO
*
        CALL  MQOPEN,                X
              (HCONN,                X
              WOD,                    X
              OPTIONS,                X
              HOBJ,                   X
              COMPCODE,               X
              REASON),VL,MF=(E,CALLLST)

*
        LA   R5,MQCC_OK              CHECK THE COMPLETION CODE
        C    R5,COMPCODE              FROM THE REQUEST AND BRANCH
        BNE  BADCALL                 TO ERROR ROUTINE IF NOT MQCC_OK

*
        MVC  TEMP_Q,WOD_OBJECTNAME   SAVE NAME OF TEMPORARY Q
*                                     CREATED BY OPEN OF MODEL Q
*
:
BADCALL DS    0H
:
*
*   CONSTANTS:
*
MOD_Q   DC   CL48'QUERY.REPLY.MODEL'  MODEL QUEUE NAME
DYN_Q   DC   CL48'QUERY.TEMPQ.*'     DYNAMIC QUEUE NAME
*
        CMQODA DSECT=NO,LIST=YES     CONSTANT VERSION OF MQOD
        CMQA   MQI VALUE EQUATES

*
*   WORKING STORAGE
*
        DFHEISTG
HCONN   DS   F                       CONNECTION HANDLE
OPTIONS DS   F                       OPEN OPTIONS
HOBJ    DS   F                       OBJECT HANDLE
COMPCODE DS F                       MQI COMPLETION CODE
REASON  DS   F                       MQI REASON CODE
TEMP_Q  DS   CL(MQ_Q_NAME_LENGTH)   SAVED QNAME AFTER OPEN
*
WOD     CMQODA DSECT=NO,LIST=YES     WORKING VERSION OF MQOD
*
CALLLST CALL ,(0,0,0,0,0,0,0,0,0,0,0),VL,MF=L LIST FORM
*                                     OF CALL
*                                     MACRO
*
:
END
```

### ***Ouverture d'une file d'attente existante***

Cet exemple montre comment utiliser l'appel MQOPEN pour ouvrir une file d'attente qui a déjà été définie.

Il montre comment spécifier deux options. Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```

:
*
*   R5 = WORK REGISTER.
*
OPEN   DS   0H
*
*       MVC  WOD_AREA,MQOD_AREA  INITIALIZE WORKING VERSION OF
*                               MQOD WITH DEFAULTS
*       MVC  WOD_OBJECTNAME,Q_NAME  SPECIFY Q NAME TO OPEN
*       LA   R5,MQOO_INPUT_EXCLUSIVE  OPEN FOR MQGET CALLS
*
*       ST   R5,OPTIONS
*
* ISSUE MQI OPEN REQUEST USING REENTRANT FORM
* OF CALL MACRO
*
*       CALL MQOPEN,                X
*           (HCONN,                  X
*            WOD,                    X
*            OPTIONS,                X
*            HOBJ,                   X
*            COMPCODE,               X
*            REASON),VL,MF=(E,CALLST)
*
*       LA   R5,MQCC_OK              CHECK THE COMPLETION CODE
*       C    R5,COMPCODE              FROM THE REQUEST AND BRANCH
*       BNE  BADCALL                 TO ERROR ROUTINE IF NOT MQCC_OK
*
*       :
BADCALL DS   0H
:
*
*   CONSTANTS:
*
Q_NAME  DC   CL48'REQUEST.QUEUE'  NAME OF QUEUE TO OPEN
*
*       CMQODA DSECT=NO,LIST=YES  CONSTANT VERSION OF MQOD
*       CMQA
*
*   WORKING STORAGE
*
*       DFHEISTG
HCONN   DS F           CONNECTION HANDLE
OPTIONS DS F           OPEN OPTIONS
HOBJ    DS F           OBJECT HANDLE
COMPCODE DS F         MQI COMPLETION CODE
REASON  DS F           MQI REASON CODE
*
WOD     CMQODA DSECT=NO,LIST=YES  WORKING VERSION OF MQOD
*
CALLLST CALL , (0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),VL,MF=L  LIST FORM
*                                     OF CALL
*                                     MACRO
*
*       :
*       END

```

### ***Fermeture d'une file d'attente***

Cet exemple montre comment utiliser l'appel MQCLOSE pour fermer une file d'attente.

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```

:
*
* ISSUE MQI CLOSE REQUEST USING REENTRANT FROM OF
* CALL MACRO
*
*       HCONN WAS SET BY A PREVIOUS MQCONN REQUEST
*       HOBJ  WAS SET BY A PREVIOUS MQOPEN REQUEST
*       R5 = WORK REGISTER
*
*       CLOSE DS   0H
*       LA   R5,MQCO_NONE           NO SPECIAL CLOSE OPTIONS

```



```

*      ST   R5,OPTIONS                ARE REQUIRED.
*
*      CALL MQCLOSE,                   X
*          (HCONN,                     X
*           HOBJ,                       X
*           OPTIONS,                    X
*           COMPCODE,                   X
*           REASON),                    X
*          VL,MF=(E,CALLLST)
*
*      LA   R5,MQCC_OK
*      C   R5,COMPCODE
*      BNE BADCALL
*
*      :
BADCALL DS   0H
*      :
*          CONSTANTS
*
*      CMQA
*
*      WORKING STORAGE (REENTRANT)
*
WEG4    DSECT
*
CALLLST CALL , (0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),VL,MF=L
*
HCONN   DS   F
HOBJ    DS   F
OPTIONS DS   F
COMPCODE DS  F
REASON  DS   F
*
*
LEG4    EQU  *-WKEG4
END

```

### ***Insertion d'un message à l'aide de MQPUT***

Cet exemple montre comment utiliser l'appel MQPUT pour placer un message dans une file d'attente.

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```

:
*      CONNECT TO QUEUE MANAGER
*
CONN    DS   0H
:
*
*      OPEN A QUEUE
*
OPEN    DS   0H
:
*
*      R4,R5,R6,R7 = WORK REGISTER.
*
PUT     DS   0H
      LA   R4,MQMD                SET UP ADDRESSES AND
      LA   R5,MQMD_LENGTH          LENGTH FOR USE BY MVCL
      LA   R6,WMD                  INSTRUCTION, AS MQMD IS
      LA   R7,WMD_LENGTH           OVER 256 BYES LONG.
      MVCL R6,R4                  INITIALIZE WORKING VERSION
*                                  OF MESSAGE DESCRIPTOR
*
*      MVC  WPMO_AREA,MQPMO_AREA   INITIALIZE WORKING MQPMO
*
*
      LA   R5,BUFFER_LEN          RETRIEVE THE BUFFER LENGTH
      ST   R5,BUFFLEN             AND SAVE IT FOR MQM USE
*
*      MVC  BUFFER,TEST_MSG        SET THE MESSAGE TO BE PUT
*
*      ISSUE MQI PUT REQUEST USING REENTRANT FORM
*      OF CALL MACRO
*
*      HCONN WAS SET BY PREVIOUS MQCONN REQUEST
*      HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST
*
      CALL MQPUT,                  X

```

```

                (HCONN,                X
                HOBJ,                   X
                WMD,                    X
                WPMO,                   X
                BUFFLEN,                 X
                BUFFER,                  X
                COMPCODE,                X
                REASON),VL,MF=(E,CALLLST)
*
        LA R5,MQCC_OK
        C  R5,COMPCODE
        BNE BADCALL
*
        :
BADCALL DS 0H
        :

```

```

*
*   CONSTANTS
*
CMQMDA DSECT=NO,LIST=YES,PERSISTENCE=MQPER_PERSISTENT
CMQPMOA DSECT=NO,LIST=YES
CMQA
TEST_MSG DC CL80'THIS IS A TEST MESSAGE'
*
*   WORKING STORAGE DSECT
*
WORKSTG DSECT
*
COMPCODE DS F
REASON   DS F
BUFFLEN  DS F
OPTIONS  DS F
HCONN    DS F
HOBJ     DS F
*
BUFFER   DS CL80
BUFFER_LEN EQU *-BUFFER
*
WMD      CMQMDA DSECT=NO,LIST=NO
WPMO     CMQPMOA DSECT=NO,LIST=NO
*
CALLLST CALL ,(0,0,0,0,0,0,0,0,0,0),VL,MF=L
*
:
:
END

```

### ***Insertion d'un message à l'aide de MQPUT1***

Cet exemple montre comment utiliser l'appel MQPUT1 pour ouvrir une file d'attente, insérer un message unique dans la file d'attente, puis fermer la file d'attente.

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```

:
*
*   CONNECT TO QUEUE MANAGER
*
CONN    DS 0H
:
*
*   R4,R5,R6,R7 = WORK REGISTER.
*
PUT     DS 0H
*
        MVC WOD_AREA,MQOD_AREA      INITIALIZE WORKING VERSION OF
*                                     MQOD WITH DEFAULTS
        MVC WOD_OBJECTNAME,Q_NAME  SPECIFY Q NAME FOR PUT1
*
        LA  R4,MQMD                  SET UP ADDRESSES AND
        LA  R5,MQMD_LENGTH           LENGTH FOR USE BY MVCL
        LA  R6,WMD                   INSTRUCTION, AS MQMD IS
        LA  R7,WMD_LENGTH            OVER 256 BYES LONG.
        MVCL R6,R4                   INITIALIZE WORKING VERSION
*                                     OF MESSAGE DESCRIPTOR

```

```

*
*      MVC  WPMO_AREA,MQPMO_AREA      INITIALIZE WORKING MQPMO
*
*      LA   R5,BUFFER_LEN             RETRIEVE THE BUFFER LENGTH
*      ST   R5,BUFFLEN                AND SAVE IT FOR MQM USE
*
*      MVC  BUFFER,TEST_MSG           SET THE MESSAGE TO BE PUT
*
* ISSUE MQI PUT REQUEST USING REENTRANT FORM OF CALL MACRO
*
*      HCONN WAS SET BY PREVIOUS MQCONN REQUEST
*      HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST
*
*      CALL MQPUT1,                   X
*      (HCONN,                        X
*       LMQOD,                        X
*       LMQMD,                        X
*       LMQPMO,                      X
*       BUFFERLENGTH,                X
*       BUFFER,                      X
*       COMPCODE,                    X
*       REASON),VL,MF=(E,CALLST)
*
*      LA   R5,MQCC_OK
*      C    R5,COMPCODE
*      BNE BADCALL
*
*      :
BADCALL DS 0H
*
*      :
*

```

```

*      CONSTANTS
*
*      CMQMDA DSECT=NO,LIST=YES,PERSISTENCE=MQPER_PERSISTENT
*      CMQPMOA DSECT=NO,LIST=YES
*      CMQODA DSECT=NO,LIST=YES
*      CMQA
*
*      TEST_MSG DC CL80'THIS IS ANOTHER TEST MESSAGE'
*      Q_NAME   DC CL48'TEST.QUEUE.NAME'
*
*      WORKING STORAGE DSECT
*
*      WORKSTG DSECT
*
*      COMPCODE DS F
*      REASON   DS F
*      BUFFLEN  DS F
*      OPTIONS  DS F
*      HCONN    DS F
*      HOBJ     DS F
*
*      BUFFER   DS CL80
*      BUFFER_LEN EQU *-BUFFER
*
*      WOD      CMQODA DSECT=NO,LIST=YES      WORKING VERSION OF MQOD
*      WMD      CMQMDA DSECT=NO,LIST=NO
*      WPMO     CMQPMOA DSECT=NO,LIST=NO
*
*      CALLST   CALL , (0,0,0,0,0,0,0,0,0,0,0,0),VL,MF=L
*
*      :
*      END

```

### ***obtention d'un message***

Cet exemple montre comment utiliser l'appel MQGET pour supprimer un message d'une file d'attente.

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```

*
*      CONNECT TO QUEUE MANAGER

```

```

*
CONN    DS   0H
:
*
*      OPEN A QUEUE FOR GET
*
OPEN    DS   0H
:
*
*      R4,R5,R6,R7 = WORK REGISTER.
*
GET     DS   0H
      LA   R4,MQMD                SET UP ADDRESSES AND
      LA   R5,MQMD_LENGTH        LENGTH FOR USE BY MVCL
      LA   R6,WMD                INSTRUCTION, AS MQMD IS
      LA   R7,WMD_LENGTH        OVER 256 BYES LONG.
      MVCL R6,R4                INITIALIZE WORKING VERSION
*                                OF MESSAGE DESCRIPTOR
*
*      MVC  WGM0_AREA,MQGM0_AREA  INITIALIZE WORKING MQGMO
*
      LA   R5,BUFFER_LEN        RETRIEVE THE BUFFER LENGTH
      ST   R5,BUFFLEN          AND SAVE IT FOR MQM USE
*
*
*      ISSUE MQI GET REQUEST USING REENTRANT FORM OF CALL MACRO
*
*      HCONN WAS SET BY PREVIOUS MQCONN REQUEST
*      HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST
*
      CALL MQGET,                X
          (HCONN,                X
           HOBJ,                  X
           WMD,                    X
           WGM0,                  X
           BUFFLEN,               X
           BUFFER,                 X
           DATALEN,              X
           COMPCODE,              X
           REASON),               X
          VL,MF=(E,CALLLST)
*
      LA   R5,MQCC_OK
      C    R5,COMPCODE
      BNE BADCALL
*
      :
BADCALL DS   0H
:

```

```

*
*      CONSTANTS
*
*      CMQMDA DSECT=NO,LIST=YES
*      CMQGMOA DSECT=NO,LIST=YES
*      CMQA
*
*      WORKING STORAGE DSECT
*
WORKSTG DSECT
*
COMPCODE DS F
REASON   DS F
BUFFLEN  DS F
DATALEN  DS F
OPTIONS  DS F
HCONN    DS F
HOBJ     DS F
*
BUFFER   DS CL80
BUFFER_LEN EQU *-BUFFER
*
WMD      CMQMDA DSECT=NO,LIST=NO
WGM0     CMQGMOA DSECT=NO,LIST=NO
*
CALLLST  CALL , (0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),VL,MF=L
*

```

```
:\nEND
```

## Obtention d'un message à l'aide de l'option d'attente

Cet exemple montre comment utiliser l'option d'attente de l'appel MQGET.

Ce code accepte les messages tronqués. Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```
:\n*\nCONNECT TO QUEUE MANAGER\nCONN DS 0H\n*\nOPEN A QUEUE FOR GET\nOPEN DS 0H\n*\nR4,R5,R6,R7 = WORK REGISTER.\nGET DS 0H\nLA R4,MQMD\nLA R5,MQMD_LENGTH\nLA R6,WMD\nLA R7,WMD_LENGTH\nMVCL R6,R4\n*\n*\nMVC WGMO_AREA,MQGMO_AREA INITIALIZE WORKING MQGMO\nL R5,=AL4(MQGMO_WAIT)\nA R5,=AL4(MQGMO_ACCEPT_TRUNCATED_MSG)\nST R5,WGMO_OPTIONS\nMVC WGMO_WAITINTERVAL,TWO_MINUTES WAIT UP TO TWO\nMINUTES BEFORE\nFALLING THE\nCALL\n*\nLA R5,BUFFER_LEN RETRIEVE THE BUFFER LENGTH\nST R5,BUFFLEN AND SAVE IT FOR MQM USE\n*\n*\nISSUE MQI GET REQUEST USING REENTRANT FORM OF CALL MACRO\n*\n*\nHCONN WAS SET BY PREVIOUS MQCONN REQUEST\nHOBJ WAS SET BY PREVIOUS MQOPEN REQUEST\n*\nCALL MQGET, X\n(HCONN, X\nHOBJ, X\nWMD, X\nWGMO, X\nBUFFLEN, X\nBUFFER, X\nDATALEN, X\nCOMPCODE, X\nREASON), X\nVL,MF=(E,CALLLST)\n*\nLA R5,MQCC_OK DID THE MQGET REQUEST\nC R5,COMPCODE WORK OK?\nBE GETOK YES, SO GO AND PROCESS.\nLA R5,MQCC_WARNING NO, SO CHECK FOR A WARNING.\nC R5,COMPCODE IS THIS A WARNING?\nBE CHECK_W YES, SO CHECK THE REASON.\n*\nLA R5,MQRC_NO_MSG_AVAILABLE IT MUST BE AN ERROR.\nIS IT DUE TO AN EMPTY\nC R5,REASON QUEUE?\nBE NOMSG YES, SO HANDLE THE ERROR\nB BADCALL NO, SO GO TO ERROR ROUTINE\n*\nCHECK_W DS 0H\nLA R5,MQRC_TRUNCATED_MSG_ACCEPTED IS THIS A\nTRUNCATED\nC R5,REASON MESSAGE?\nBE GETOK YES, SO GO AND PROCESS.\nB BADCALL NO, SOME OTHER WARNING\n*\nNOMSG DS 0H
```

```

:
GETOK DS 0H
:

```

```

BADCALL DS 0H
:
*
*   CONSTANTS
*
*   CMQMDA DSECT=NO,LIST=YES
*   CMQMOA DSECT=NO,LIST=YES
*   CMQA
*
* TWO_MINUTES DC F'120000'      GET WAIT INTERVAL
*
*   WORKING STORAGE DSECT

```

```

*
WORKSTG DSECT
*
COMPCODE DS F
REASON DS F
BUFFLEN DS F
DATALEN DS F
OPTIONS DS F
HCONN DS F
HOBJ DS F
*
BUFFER DS CL80
BUFFER_LEN EQU *-BUFFER
*
WMD CMQMDA DSECT=NO,LIST=NO
WGMO CMQMOA DSECT=NO,LIST=NO
*
CALLLST CALL ,(0,0,0,0,0,0,0,0,0,0),VL,MF=L
*
:
END

```

### ***Obtention d'un message à l'aide de la notification***

Cet exemple montre comment utiliser l'appel MQGET pour définir un signal afin d'être averti lorsqu'un message approprié arrive dans une file d'attente.

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```

:
*
*   CONNECT TO QUEUE MANAGER
*
CONN DS 0H
:
*
*   OPEN A QUEUE FOR GET
*
OPEN DS 0H
:
*
*   R4,R5,R6,R7 = WORK REGISTER.
*
GET DS 0H
LA R4,MQMD          SET UP ADDRESSES AND
LA R5,MQMD_LENGTH  LENGTH FOR USE BY MVCL
LA R6,WMD          INSTRUCTION, AS MQMD IS
LA R7,WMD_LENGTH  OVER 256 BYES LONG.
MVCL R6,R4        INITIALIZE WORKING VERSION
*                  OF MESSAGE DESCRIPTOR

```

```

*
MVC WGMO_AREA,MQGMO_AREA  INITIALIZE WORKING MQGMO
LA R5,MQGMO_SET_SIGNAL
ST R5,WGMO_OPTIONS
MVC WGMO_WAITINTERVAL,FIVE_MINUTES  WAIT UP TO FIVE

```

```

*                                     MINUTES BEFORE
*                                     FAILING THE CALL
*
XC  SIG_ECB,SIG_ECB  CLEAR THE ECB
LA  R5,SIG_ECB      GET THE ADDRESS OF THE ECB
ST  R5,WGMO_SIGNAL1 AND PUT IT IN THE WORKING
*                                     MQGMO
*
LA  R5,BUFFER_LEN   RETRIEVE THE BUFFER LENGTH
ST  R5,BUFFLEN      AND SAVE IT FOR MQM USE
*
*
*   ISSUE MQI GET REQUEST USING REENTRANT FORM OF CALL MACRO
*
HCONN WAS SET BY PREVIOUS MQCONN REQUEST
HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST
*
CALL  MQGET,          X
      (HCONN,         X
       HOBJ,          X
       WMD,           X
       WGMO,          X
       BUFFLEN,       X
       BUFFER,        X
       DATALEN,      X
       COMPCODE,      X
       REASON),       X
      VL,MF=(E,CALLST)
*
LA  R5,MQCC_OK       DID THE MQGET REQUEST
C   R5,COMPCODE      WORK OK?
BE  GETOK            YES, SO GO AND PROCESS.
LA  R5,MQCC_WARNING  NO, SO CHECK FOR A WARNING.
C   R5,COMPCODE      IS THIS A WARNING?
BE  CHECK_W          YES, SO CHECK THE REASON.
B   BADCALL          NO, SO GO TO ERROR ROUTINE
*

```

```

CHECK_W DS 0H
LA R5,MQRC_SIGNAL_REQUEST_ACCEPTED
C R5,REASON  SIGNAL REQUEST SIGNAL SET?
BNE BADCALL NO, SOME ERROR OCCURRED
B DOWORK    YES, SO DO SOMETHING
*          ELSE
*
CHECKSIG DS 0H
CLC SIG_ECB+1(3),=AL3(MQEC_MSG_ARRIVED)
*          IS A MESSAGE AVAILABLE?
BE GET      YES, SO GO AND GET IT
*
CLC SIG_ECB+1(3),=AL3(MQEC_WAIT_INTERVAL_EXPIRED)
*          HAVE WE WAITED LONG ENOUGH?
BE NOMSG    YES, SO SAY NO MSG AVAILABLE
B BADCALL   IF IT'S ANYTHING ELSE
*          GO TO ERROR ROUTINE.
*
DOWORK DS 0H
*
*   TM SIG_ECB,X'40'  HAS THE SIGNAL ECB BEEN POSTED?
BO CHECKSIG YES, SO GO AND CHECK WHY
B DOWORK    NO, SO GO AND DO MORE WORK
*
NOMSG DS 0H
*
GETOK DS 0H
*
BADCALL DS 0H
*
*   CONSTANTS
*
CMQMDA DSECT=NO,LIST=YES
CMQMOA DSECT=NO,LIST=YES
CMQA
*
FIVE_MINUTES DC F'300000'  GET SIGNAL INTERVAL
*
*   WORKING STORAGE DSECT
*

```

```

WORKSTG DSECT
*
COMPCODE DS F
REASON DS F
BUFFLEN DS F
DATALEN DS F
OPTIONS DS F
HCONN DS F
HOBJ DS F
SIG_ECB DS F

```

```

*
BUFFER DS CL80
BUFFER_LEN EQU *-BUFFER
*
WMD CMQMDA DSECT=NO,LIST=NO
WGMO CMQGMOA DSECT=NO,LIST=NO
*
CALLLIST CALL ,(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),VL,MF=L
*
:
END

```

### ***Interrogation et définition des attributs d'une file d'attente***

Cet exemple montre comment utiliser l'appel MQINQ pour consulter les attributs d'une file d'attente et utiliser l'appel MQSET pour modifier les attributs d'une file d'attente.

Cette extraction est extraite de l'exemple d'application Attributs de file d'attente (programme CSQ4CAC1) fourni avec IBM MQ for z/OS.

```

:
DFHEISTG DSECT
:
OBJDESC CMQODA LIST=YES Working object descriptor
*
SELECTORCOUNT DS F Number of selectors
INTATTRCOUNT DS F Number of integer attributes
CHARATTRLENGTH DS F char attributes length
CHARATTRS DS C Area for char attributes
*
OPTIONS DS F Command options
HCONN DS F Handle of connection
HOBJ DS F Handle of object
COMPCODE DS F Completion code
REASON DS F Reason code
SELECTOR DS 2F Array of selectors
INTATTRS DS 2F Array of integer attributes
:
OBJECT DS CL(MQ_Q_NAME_LENGTH) Name of queue
:
CALLLIST CALL ,(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),VL,MF=L
*****
* PROGRAM EXECUTION STARTS HERE *
:
CSQ4CAC1 DFHEIENT CODEREG=(R3),DATAREG=(R13)
:
* Initialize the variables for the set call
*
SR R0,R0 Clear register zero
ST R0,CHARATTRLENGTH Set char length to zero
LA R0,2 Load to set
ST R0,SELECTORCOUNT selectors add
ST R0,INTATTRCOUNT integer attributes
*
LA R0,MQIA_INHIBIT_GET Load q attribute selector
ST R0,SELECTOR+0 Place in field
LA R0,MQIA_INHIBIT_PUT Load q attribute selector
ST R0,SELECTOR+4 Place in field
*
UPDTEST DS 0H
CLC ACTION,CINHIB Are we inhibiting?
BE UPDINHBT Yes branch to section
*
CLC ACTION,CALLOW Are we allowing?
BE UPDALLOW Yes branch to section

```



```

*
MVC M00_MSG,M01_MSG1 Invalid request
BR R6 Return to caller
*

```

```

UPDINHBT DS 0H
MVC UPDTYPE,CINHIBIT Indicate action type
LA R0,MQQA_GET_INHIBITED Load attribute value
ST R0,INTATTRS+0 Place in field
LA R0,MQQA_PUT_INHIBITED Load attribute value
ST R0,INTATTRS+4 Place in field
B UPDCALL Go and do call

```

```

*
UPDALLOW DS 0H
MVC UPDTYPE,CALLOWED Indicate action type
LA R0,MQQA_GET_ALLOWED Load attribute value
ST R0,INTATTRS+0 Place in field
LA R0,MQQA_PUT_ALLOWED Load attribute value
ST R0,INTATTRS+4 Place in field
B UPDCALL Go and do call

```

```

*
UPDCALL DS 0H
CALL MQSET, C
(HCONN, C
HOBJ, C
SELECTORCOUNT, C
SELECTOR, C
INTATTRCOUNT, C
INTATTRS, C
CHARATTRLENGTH, C
CHARATTRS, C
COMPCODE, C
REASON), C
VL,MF=(E,CALLLIST)

```

```

*
LA R0,MQCC_OK Load expected compcode
C R0,COMPCODE Was set successful?
:

```

```

* SECTION NAME : INQUIRE *
* FUNCTION : Inquires on the objects attributes *
* CALLED BY : PROCESS *
* CALLS : OPEN, CLOSE, CODES *
* RETURN : To Register 6 *
INQUIRE DS 0H
:

```

```

* Initialize the variables for the inquire call

```

```

*
SR R0,R0 Clear register zero
ST R0,CHARATTRLENGTH Set char length to zero
LA R0,2 Load to set
ST R0,SELECTORCOUNT selectors add
ST R0,INTATTRCOUNT integer attributes

```

```

*
LA R0,MQIA_INHIBIT_GET Load attribute value
ST R0,SELECTOR+0 Place in field
LA R0,MQIA_INHIBIT_PUT Load attribute value
ST R0,SELECTOR+4 Place in field
CALL MQINQ, C
(HCONN, C
HOBJ, C
SELECTORCOUNT, C
SELECTOR, C
INTATTRCOUNT, C
INTATTRS, C
CHARATTRLENGTH, C
CHARATTRS, C
COMPCODE, C
REASON), C
VL,MF=(E,CALLLIST)
LA R0,MQCC_OK Load expected compcode
C R0,COMPCODE Was inquire successful?
:

```

## Exemples PL/I

L'utilisation de PL/I est prise en charge par z/OS uniquement. Cette collection de rubriques illustre des techniques utilisant des exemples PL/I.

### Connexion à un gestionnaire de files d'attente

Cet exemple montre comment utiliser l'appel MQCONN pour connecter un programme à un gestionnaire de files d'attente dans z/OS batch.

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```
%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****
/* STRUCTURE BASED ON PARAMETER INPUT AREA (PARAM) */
*****/
DCL 1 INPUT_PARAM      BASED(ADDR(PARAM)),
    2 PARAM_LENGTH     FIXED BIN(15),
    2 PARAM_MQMNNAME   CHAR(48);
:

/*****
/* WORKING STORAGE DECLARATIONS */
*****/
DCL MQMNNAME           CHAR(48);
DCL COMPCODE           BINARY FIXED (31);
DCL REASON             BINARY FIXED (31);
DCL HCONN             BINARY FIXED (31);
:

/*****
/* COPY QUEUE MANAGER NAME PARAMETER */
/* TO LOCAL STORAGE */
*****/
MQMNNAME = ' ';
MQMNNAME = SUBSTR(PARAM_MQMNNAME,1,PARAM_LENGTH);
:

/*****
/* CONNECT FROM THE QUEUE MANAGER */
*****/
CALL MQCONN (MQMNNAME, /* MQM SYSTEM NAME */
            HCONN, /* CONNECTION HANDLE */
            COMPCODE, /* COMPLETION CODE */
            REASON); /* REASON CODE */

/*****
/* TEST THE COMPLETION CODE OF THE CONNECT CALL. */
/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE */
/* SHOWING THE COMPLETION CODE AND THE REASON CODE. */
*****/
IF COMPCODE = MQCC_OK
    THEN DO;
:
    CALL ERROR_ROUTINE;
END;
```

### Déconnexion d'un gestionnaire de files d'attente

Cet exemple montre comment utiliser l'appel MQDISC pour déconnecter un programme d'un gestionnaire de files d'attente dans un lot z/OS .

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```
%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****
/* WORKING STORAGE DECLARATIONS */
*****/
DCL COMPCODE           BINARY FIXED (31);
DCL REASON             BINARY FIXED (31);
DCL HCONN             BINARY FIXED (31);
:

/*****
/* DISCONNECT FROM THE QUEUE MANAGER */
*****/
```

```

/*****/
CALL MQDISC (HCONN,      /* CONNECTION HANDLE */
             COMPCODE,   /* COMPLETION CODE   */
             REASON);    /* REASON CODE       */
/*****/

/*****/
/* TEST THE COMPLETION CODE OF THE DISCONNECT CALL. */
/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE   */
/* SHOWING THE COMPLETION CODE AND THE REASON CODE. */
/*****/
IF COMPCODE /= MQCC_OK
  THEN DO;
  :
  CALL ERROR_ROUTINE;
END;

```

### ***Création d'une file d'attente dynamique***

Cet exemple montre comment utiliser l'appel MQOPEN pour créer une file d'attente dynamique.

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```

%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****/
/* WORKING STORAGE DECLARATIONS */
/*****/
DCL COMPCODE          BINARY FIXED (31);
DCL REASON            BINARY FIXED (31);
DCL HCONN             BINARY FIXED (31);
DCL HOBJ              BINARY FIXED (31);
DCL OPTIONS           BINARY FIXED (31);
:
DCL MODEL_QUEUE_NAME CHAR(48) INIT('PL1.REPLY.MODEL');
DCL DYNAMIC_NAME_PREFIX CHAR(48) INIT('PL1.TEMPQ.*');
DCL DYNAMIC_QUEUE_NAME CHAR(48) INIT(' ');
:
/*****/
/* LOCAL COPY OF OBJECT DESCRIPTOR */
/*****/
DCL 1 LMQOD LIKE MQOD;
:
/*****/
/* SET UP OBJECT DESCRIPTOR FOR OPEN OF REPLY QUEUE */
/*****/
LMQOD.OBJECTTYPE =MQOT_Q;
LMQOD.OBJECTNAME = MODEL_QUEUE_NAME;
LMQOD.DYNAMICQNAME = DYNAMIC_NAME_PREFIX;
OPTIONS = MQOO_INPUT_EXCLUSIVE;

CALL MQOPEN (HCONN,
             LMQOD,
             OPTIONS,
             HOBJ,
             COMPCODE,
             REASON);

/*****/
/* TEST THE COMPLETION CODE OF THE OPEN CALL. */
/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE */
/* SHOWING THE COMPLETION CODE AND THE REASON CODE. */
/* IF THE CALL HAS SUCCEEDED THEN EXTRACT THE NAME OF */
/* THE NEWLY CREATED DYNAMIC QUEUE FROM THE OBJECT */
/* DESCRIPTOR. */
/*****/
IF COMPCODE /= MQCC_OK
  THEN DO;
  :
  CALL ERROR_ROUTINE;
END;
ELSE
  DYNAMIC_QUEUE_NAME = LMQOD_OBJECTNAME;

```

### ***Ouverture d'une file d'attente existante***

Cet exemple montre comment utiliser l'appel MQOPEN pour ouvrir une file d'attente existante.

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```
%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****/
/* WORKING STORAGE DECLARATIONS */
/*****/
DCL COMPCODE          BINARY FIXED (31);
DCL REASON            BINARY FIXED (31);
DCL HCONN            BINARY FIXED (31);
DCL HOBJ            BINARY FIXED (31);
DCL OPTIONS          BINARY FIXED (31);
:
DCL QUEUE_NAME        CHAR(48) INIT('PL1.LOCAL.QUEUE');
:
/*****/
/* LOCAL COPY OF OBJECT DESCRIPTOR */
/*****/
DCL 1 LMQOD LIKE MQOD;
:
/*****/
/* SET UP OBJECT DESCRIPTOR FOR OPEN OF REPLY QUEUE */
/*****/
LMQOD.OBJECTTYPE = MQOT_Q;
LMQOD.OBJECTNAME = QUEUE_NAME;
OPTIONS = MQOO_INPUT_EXCLUSIVE;

CALL MQOPEN (HCONN,
             LMQOD,
             OPTIONS,
             HOBJ,
             COMPCODE,
             REASON);

/*****/
/* TEST THE COMPLETION CODE OF THE OPEN CALL. */
/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE */
/* SHOWING THE COMPLETION CODE AND THE REASON CODE. */
/*****/
IF COMPCODE /= MQCC_OK
  THEN DO;
  :
  CALL ERROR_ROUTINE;
END;
```

### ***Fermeture d'une file d'attente***

Cet exemple montre comment utiliser l'appel MQCLOSE.

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```
%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****/
/* WORKING STORAGE DECLARATIONS */
/*****/
DCL COMPCODE          BINARY FIXED (31);
DCL REASON            BINARY FIXED (31);
DCL HCONN            BINARY FIXED (31);
DCL HOBJ            BINARY FIXED (31);
DCL OPTIONS          BINARY FIXED (31);
:
/*****/
/* SET CLOSE OPTIONS */
/*****/
OPTIONS=MQCO_NONE;

/*****/
/* CLOSE QUEUE */
/*****/
CALL MQCLOSE (HCONN, /* CONNECTION HANDLE */
             HOBJ, /* OBJECT HANDLE */
             OPTIONS, /* CLOSE OPTIONS */
             COMPCODE, /* COMPLETION CODE */
             REASON); /* REASON CODE */
```

```

/*****/
/* TEST THE COMPLETION CODE OF THE CLOSE CALL. */
/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE */
/* SHOWING THE COMPLETION CODE AND THE REASON CODE. */
/*****/
IF COMPCODE = MQCC_OK
  THEN DO;
  :
  CALL ERROR_ROUTINE;
END;

```

## ***Insertion d'un message à l'aide de MQPUT***

Cet exemple montre comment utiliser l'appel MQPUT à l'aide du contexte.

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```

%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****/
/* WORKING STORAGE DECLARATIONS */
/*****/
DCL COMPCODE          BINARY FIXED (31);
DCL REASON            BINARY FIXED (31);
DCL HCONN             BINARY FIXED (31);
DCL HOBJ              BINARY FIXED (31);
DCL OPTIONS           BINARY FIXED (31);
DCL BUFFLEN          BINARY FIXED (31);
DCL BUFFER            CHAR(80);
:
DCL PL1_TEST_MESSAGE CHAR(80)
INIT('***** THIS IS A TEST MESSAGE *****');
:
/*****/
/* LOCAL COPY OF MESSAGE DESCRIPTOR */
/* AND PUT MESSAGE OPTIONS */
/*****/
DCL 1 LMQMD LIKE MQMD;
DCL 1 LMQPMO LIKE MQPMO;
:
/*****/
/* SET UP MESSAGE DESCRIPTOR */
/*****/
LMQMD.MSGTYPE = MQMT_DATAGRAM;
LMQMD.PRIORITY = 1;
LMQMD.PERSISTENCE = MQPER_PERSISTENT;
LMQMD.REPLYTOQ = ' ';
LMQMD.REPLYTOQMGR = ' ';
LMQMD.MSGID = MQMI_NONE;
LMQMD.CORRELID = MQCI_NONE;

/*****/
/* SET UP PUT MESSAGE OPTIONS */
/*****/
LMQPMO.OPTIONS = MQPMO_NO_SYNCPOINT;

/*****/
/* SET UP LENGTH OF MESSAGE BUFFER AND THE MESSAGE */
/*****/
BUFFLEN = LENGTH(BUFFER);
BUFFER = PL1_TEST_MESSAGE;
/*****/
/*
*/
/* HCONN WAS SET BY PREVIOUS MQCONN REQUEST. */
/* HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST. */
/*
*/
/*****/
CALL MQPUT (HCONN,
           HOBJ,
           LMQMD,
           LMQPMO,
           BUFFLEN,
           BUFFER,
           COMPCODE,
           REASON);

```

```

/*****/
/* TEST THE COMPLETION CODE OF THE PUT CALL.          */
/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE      */
/* SHOWING THE COMPLETION CODE AND THE REASON CODE.   */
/*****/
      IF COMPCODE = MQCC_OK
      THEN DO;
        :
        :
        CALL ERROR_ROUTINE;
      END;

```

## ***Insertion d'un message à l'aide de MQPUT1***

Cet exemple montre comment utiliser l'appel MQPUT1 .

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```

%INCLUDE SYSLIB(CMQEPP);
%INCLUDE SYSLIB(CMQP);
:
/*****/
/* WORKING STORAGE DECLARATIONS                      */
/*****/
DCL COMPCODE      BINARY FIXED (31);
DCL REASON        BINARY FIXED (31);
DCL HCONN         BINARY FIXED (31);
DCL OPTIONS       BINARY FIXED (31);
DCL BUFFLEN       BINARY FIXED (31);
DCL BUFFER        CHAR(80);
:
DCL REPLY_TO_QUEUE CHAR(48) INIT('PL1.REPLY.QUEUE');
DCL QUEUE_NAME     CHAR(48) INIT('PL1.LOCAL.QUEUE');
DCL PL1_TEST_MESSAGE CHAR(80)
INIT('***** THIS IS ANOTHER TEST MESSAGE *****');
:
/*****/
/* LOCAL COPY OF OBJECT DESCRIPTOR, MESSAGE DESCRIPTOR */
/* AND PUT MESSAGE OPTIONS                               */
/*****/
DCL 1 LMQOD LIKE MQOD;
DCL 1 LMQMD LIKE MQMD;
DCL 1 LMQPMO LIKE MQPMO;
:
/*****/
/* SET UP OBJECT DESCRIPTOR AS REQUIRED.                */
/*****/
LMQOD.OBJECTTYPE = MQOT_Q;
LMQOD.OBJECTNAME = QUEUE_NAME;

/*****/
/* SET UP MESSAGE DESCRIPTOR AS REQUIRED.              */
/*****/
LMQMD.MSGTYPE = MQMT_REQUEST;
LMQMD.PRIORITY = 5;
LMQMD.PERSISTENCE = MQPER_PERSISTENT;
LMQMD.REPLYTOQ = REPLY_TO_QUEUE;
LMQMD.REPLYTOQMGR = 'T';
LMQMD.MSGID = MQMI_NONE;
LMQMD.CORRELID = MQCI_NONE;

/*****/
/* SET UP PUT MESSAGE OPTIONS AS REQUIRED                */
/*****/
LMQPMO.OPTIONS = MQPMO_NO_SYNCPOINT;

/*****/
/* SET UP LENGTH OF MESSAGE BUFFER AND THE MESSAGE    */
/*****/
BUFFLEN = LENGTH(BUFFER);
BUFFER = PL1_TEST_MESSAGE;

CALL MQPUT1 (HCONN,
             LMQOD,
             LMQMD,
             LMQPMO,

```

```

BUFFLEN,
BUFFER,
COMPCODE,
REASON);

/*****/
/* TEST THE COMPLETION CODE OF THE PUT1 CALL. */
/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE SHOWING */
/* THE COMPLETION CODE AND THE REASON CODE. */
/*****/
IF COMPCODE = MQCC_OK
THEN DO;
:
CALL ERROR_ROUTINE;
END;

```

### ***obtention d'un message***

Cet exemple montre comment utiliser l'appel MQGET pour supprimer un message d'une file d'attente.

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```

%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****/
/* WORKING STORAGE DECLARATIONS */
/*****/
DCL COMPCODE          BINARY FIXED (31);
DCL REASON            BINARY FIXED (31);
DCL HCONN             BINARY FIXED (31);
DCL HOBJ              BINARY FIXED (31);
DCL BUFFLEN          BINARY FIXED (31);
DCL DATALEN         BINARY FIXED (31);
DCL BUFFER            CHAR(80);
:

/*****/
/* LOCAL COPY OF MESSAGE DESCRIPTOR AND */
/* GET MESSAGE OPTIONS */
/*****/
DCL 1 LMQMD LIKE MQMD;
DCL 1 LMQGMO LIKE MQGMO;
:
/*****/
/* SET UP MESSAGE DESCRIPTOR AS REQUIRED. */
/* MSGID AND CORRELID IN MQMD SET TO NULLS SO FIRST */
/* AVAILABLE MESSAGE WILL BE RETRIEVED. */
/*****/
LMQMD.MSGID = MQMI_NONE;
LMQMD.CORRELID = MQCI_NONE;

/*****/
/* SET UP GET MESSAGE OPTIONS AS REQUIRED. */
/*****/
LMQGMO.OPTIONS = MQGMO_NO_SYNCPOINT;

/*****/
/* SET UP LENGTH OF MESSAGE BUFFER. */
/*****/
BUFFLEN = LENGTH(BUFFER);
/*****/
/*
/* HCONN WAS SET BY PREVIOUS MQCONN REQUEST. */
/* HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST. */
/*
/*****/

CALL MQGET (HCONN,
           HOBJ,
           LMQMD,
           LMQGMO,
           BUFFERLEN,
           BUFFER,
           DATALEN,
           COMPCODE,
           REASON);

```

```

/*****/
/* TEST THE COMPLETION CODE OF THE GET CALL. */
/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE */
/* SHOWING THE COMPLETION CODE AND THE REASON CODE. */
/*****/
IF COMPCODE = MQCC_OK
  THEN DO;
  :
  CALL ERROR_ROUTINE;
END;

```

## Obtention d'un message à l'aide de l'option d'attente

Cet exemple montre comment utiliser l'appel MQGET avec l'option d'attente et accepter les messages tronqués.

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```

%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****/
/* WORKING STORAGE DECLARATIONS */
/*****/
DCL COMPCODE          BINARY FIXED (31);
DCL REASON            BINARY FIXED (31);
DCL HCONN             BINARY FIXED (31);
DCL HOBJ              BINARY FIXED (31);
DCL BUFFLEN          BINARY FIXED (31);
DCL DATALEN         BINARY FIXED (31);
DCL BUFFER            CHAR(80);
:
/*****/
/* LOCAL COPY OF MESSAGE DESCRIPTOR AND GET MESSAGE */
/* OPTIONS */
/*****/
DCL 1 LMQMD LIKE MQMD;
DCL 1 LMQGMO LIKE MQGMO;
:
/*****/
/* SET UP MESSAGE DESCRIPTOR AS REQUIRED. */
/* MSGID AND CORRELID IN MQMD SET TO NULLS SO FIRST */
/* AVAILABLE MESSAGE WILL BE RETRIEVED. */
/*****/
LMQMD.MSGID = MQMI_NONE;
LMQMD.CORRELID = MQCI_NONE;

/*****/
/* SET UP GET MESSAGE OPTIONS AS REQUIRED. */
/* WAIT INTERVAL SET TO ONE MINUTE. */
/*****/
LMQGMO.OPTIONS = MQGMO_WAIT +
                 MQGMO_ACCEPT_TRUNCATED_MSG +
                 MQGMO_NO_SYNCPOINT;
LMQGMO.WAITINTERVAL=60000;

/*****/
/* SET UP LENGTH OF MESSAGE BUFFER. */
/*****/
BUFFLEN = LENGTH(BUFFER);

/*****/
/*
/* HCONN WAS SET BY PREVIOUS MQCONN REQUEST. */
/* HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST. */
/*
/*****/

CALL MQGET (HCONN,
            HOBJ,
            LMQMD,
            LMQGMO,
            BUFFERLEN,
            BUFFER,
            DATALEN,
            COMPCODE,

```



```

REASON);

/*****
/* TEST THE COMPLETION CODE OF THE GET CALL.          */
/* TAKE APPROPRIATE ACTION BASED ON COMPLETION CODE AND */
/* REASON CODE.                                       */
*****/

SELECT (COMPCODE);
  WHEN (MQCC_OK) DO; /* GET WAS SUCCESSFUL */
  :
  END;
  WHEN (MQCC_WARNING) DO;
  IF REASON = MQRC_TRUNCATED_MSG_ACCEPTED
  THEN DO; /* GET WAS SUCCESSFUL */
  :
  END;
  ELSE DO;
  :
  CALL ERROR_ROUTINE;
  END;
  END;
  WHEN (MQCC_FAILED) DO;
  :
  CALL ERROR_ROUTINE;
  END;
  END;
  OTHERWISE;
END;

```

### ***Obtention d'un message à l'aide de la notification***

Extrait de code qui explique comment utiliser l'appel MQGET avec la notification.

**La signalisation est disponible uniquement avec IBM MQ for z/OS .**

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```

%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****
/* WORKING STORAGE DECLARATIONS          */
*****/
DCL COMPCODE          BINARY FIXED (31);
DCL REASON            BINARY FIXED (31);
DCL HCONN             BINARY FIXED (31);
DCL HOBJ              BINARY FIXED (31);
DCL DATALEN         BINARY FIXED (31);
DCL BUFFLEN          BINARY FIXED (31);
DCL BUFFER           CHAR(80);
:
DCL ECB_FIXED        FIXED BIN(31);
DCL 1 ECB_OVERLAY   BASED(ADDR(ECB_FIXED)),
      3 ECB_WAIT    BIT,
      3 ECB_POSTED  BIT,
      3 ECB_FLAG3_8 BIT(6),
      3 ECB_CODE    PIC'999';
:
/*****
/* LOCAL COPY OF MESSAGE DESCRIPTOR AND GET MESSAGE */
/* OPTIONS                                           */
*****/
DCL 1 LMQMD LIKE MQMD;
DCL 1 LMQGMO LIKE MQGMO;
:
/*****
/* CLEAR ECB FIELD.                                */
*****/
ECB_FIXED = 0;
:
/*****
/* SET UP MESSAGE DESCRIPTOR AS REQUIRED.          */
/* MSGID AND CORRELID IN MQMD SET TO NULLS SO FIRST */
/* AVAILABLE MESSAGE WILL BE RETRIEVED.          */
*****/
LMQMD.MSGID = MQMI_NONE;
LMQMD.CORRELID = MQCI_NONE;
/*****

```

```

/* SET UP GET MESSAGE OPTIONS AS REQUIRED.          */
/* WAIT INTERVAL SET TO ONE MINUTE.                */
/*****
  LMQGMO.OPTIONS = MQGMO_SET_SIGNAL +
                  MQGMO_NO_SYNCPOINT;
  LMQGMO.WAITINTERVAL=60000;
  LMQGMO.SIGNAL1 = ADDR(ECB_FIXED);
*****/

```

```

/*****
/* SET UP LENGTH OF MESSAGE BUFFER.                */
/* CALL MESSAGE RETRIEVAL ROUTINE.                 */
/*****
  BUFFLEN = LENGTH(BUFFER);
  CALL GET_MSG;
*****/

```

```

/*****
/* TEST THE COMPLETION CODE OF THE GET CALL.       */
/* TAKE APPROPRIATE ACTION BASED ON COMPLETION CODE AND REASON CODE.
/*****

```

```

SELECT;
  WHEN ((COMPCODE = MQCC_OK) &
        (REASON = MQCC_NONE)) DO
    :
    CALL MSG_ROUTINE;
    :
  END;
  WHEN ((COMPCODE = MQCC_WARNING) &
        (REASON = MQRC_SIGNAL_REQUEST_ACCEPTED)) DO;
    :
    CALL DO_WORK;
    :
  END;
  WHEN ((COMPCODE = MQCC_FAILED) &
        (REASON = MQRC_SIGNAL_OUTSTANDING)) DO;
    :
    CALL DO_WORK;
    :
  END;
  OTHERWISE DO;          /* FAILURE CASE */
/*****
/* ISSUE AN ERROR MESSAGE SHOWING THE COMPLETION CODE
/* AND THE REASON CODE.
/*****
  :
  CALL ERROR_ROUTINE;
  :
  END;
END;
:

```

```

DO_WORK: PROC;
:
  IF ECB_POSTED
  THEN DO;
    SELECT(ECB_CODE);
    WHEN(MQEC_MSG_ARRIVED) DO;
      :
      CALL GET_MSG;
      :
    END;
    WHEN(MQEC_WAIT_INTERVAL_EXPIRED) DO;
      :
      CALL NO_MSG;
      :
    END;
    OTHERWISE DO;          /* FAILURE CASE */
/*****
/* ISSUE AN ERROR MESSAGE SHOWING THE COMPLETION CODE
/* AND THE REASON CODE.
/*****
  :
  CALL ERROR_ROUTINE;
  :
  END;

```

```

        END;
        END;
        :
END DO_WORK;
GET_MSG: PROC;

```

```

/*****/
/*
/* HCONN WAS SET BY PREVIOUS MQCONN REQUEST.          */
/* HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST.          */
/* MD AND GMO SET UP AS REQUIRED.                    */
/*
/*****/

        CALL MQGET (HCONN,
                    HOBJ,
                    LMQMD,
                    LMQGMO,
                    BUFLLEN,
                    BUFFER,
                    DATALEN,
                    COMPCODE,
                    REASON);

END GET_MSG;

NO_MSG: PROC;
:
END NO_MSG;

```

### ***Interrogation des attributs d'un objet***

Cet exemple montre comment utiliser l'appel MQINQ pour demander les attributs d'une file d'attente.

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```

%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****/
/* WORKING STORAGE DECLARATIONS          */
/*****/
DCL COMPCODE          BINARY FIXED (31);
DCL REASON            BINARY FIXED (31);
DCL HCONN             BINARY FIXED (31);
DCL HOBJ             BINARY FIXED (31);
DCL OPTIONS           BINARY FIXED (31);
DCL SELECTORCOUNT   BINARY FIXED (31);
DCL INTATTRCOUNT    BINARY FIXED (31);
DCL 1 SELECTOR_TABLE,
    3 SELECTORS(5)    BINARY FIXED (31);
DCL 1 INTATTR_TABLE,
    3 INTATTRS(5)    BINARY FIXED (31);
DCL CHARATTRLENGTH   BINARY FIXED (31);
DCL CHARATTRS        CHAR(100);
:

/*****/
/* SET VARIABLES FOR INQUIRE CALL        */
/* INQUIRE ON THE CURRENT QUEUE DEPTH    */
/*****/

        SELECTORS(01) = MQIA_CURRENT_Q_DEPTH;

        SELECTORCOUNT = 1;
        INTATTRCOUNT = 1;

        CHARATTRLENGTH = 0;
/*****/
/*
/* HCONN WAS SET BY PREVIOUS MQCONN REQUEST.          */
/* HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST.          */
/*
/*****/
        CALL MQINQ (HCONN,

```

```

HOBJ,
SELECTORCOUNT,
SELECTORS,
INTATTRCOUNT,
INTATTRS,
CHARATTRLENGTH,
CHARATTRS,
COMPCODE,
REASON);

```

```

/*****/
/* TEST THE COMPLETION CODE OF THE INQUIRE CALL. */
/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE SHOWING */
/* THE COMPLETION CODE AND THE REASON CODE. */
/*****/
IF COMPCODE = MQCC_OK
THEN DO;
:
CALL ERROR_ROUTINE;
END;

```

### ***Définition des attributs d'une file d'attente***

Cet exemple montre comment utiliser l'appel MQSET pour modifier les attributs d'une file d'attente.

Cet extrait n'est pas extrait des exemples d'application fournis avec IBM MQ.

```

%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****/
/* WORKING STORAGE DECLARATIONS */
/*****/
DCL COMPCODE          BINARY FIXED (31);
DCL REASON            BINARY FIXED (31);
DCL HCONN            BINARY FIXED (31);
DCL HOBJ            BINARY FIXED (31);
DCL OPTIONS          BINARY FIXED (31);
DCL SELECTORCOUNT  BINARY FIXED (31);
DCL INTATTRCOUNT  BINARY FIXED (31);
DCL 1 SELECTOR_TABLE,
   3 SELECTORS(5)      BINARY FIXED (31);
DCL 1 INTATTR_TABLE,
   3 INTATTRS(5)      BINARY FIXED (31);
DCL CHARATTRLENGTH  BINARY FIXED (31);
DCL CHARATTRS       CHAR(100);
:

/*****/
/* SET VARIABLES FOR SET CALL */
/* SET GET AND PUT INHIBITED */
/*****/

SELECTORS(01) = MQIA_INHIBIT_GET;
SELECTORS(02) = MQIA_INHIBIT_PUT;

INTATTRS(01) = MQQA_GET_INHIBITED;
INTATTRS(02) = MQQA_PUT_INHIBITED;

SELECTORCOUNT = 2;
INTATTRCOUNT = 2;

CHARATTRLENGTH = 0;

/*****/
/*
/* HCONN WAS SET BY PREVIOUS MQCONN REQUEST. */
/* HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST. */
/*
/*****/
CALL MQSET (HCONN,
           HOBJ,
           SELECTORCOUNT,
           SELECTORS,
           INTATTRCOUNT,

```

```

        INTATTRS,
        CHARATTRLENGTH,
        CHARATTRS,
        COMPCODE,
        REASON);

/*****
/* TEST THE COMPLETION CODE OF THE SET CALL.          */
/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE SHOWING */
/* THE COMPLETION CODE AND THE REASON CODE.          */
*****/
        IF COMPCODE /= MQCC_OK
            THEN DO;
            :
            CALL ERROR_ROUTINE;
        END;

```

## Constantes

Utilisez les informations de référence présentées dans cette section pour accomplir les tâches adaptées à vos besoins métier.

## Fichiers COPY, d'en-tête, d'inclusion et de module IBM MQ

Ces informations sont des informations générales sur l'interface de programmation.

Cette section contient des informations pour vous aider à utiliser l'interface MQI pour différents langages de programmation, comme suit.

### Fichiers d'en-tête C

Des fichiers d'en-tête sont fournis pour vous aider à écrire des programmes d'application C qui utilisent l'interface MQI.

Les fichiers d'en-tête C sont récapitulés dans le tableau suivant:

<i>Tableau 1. Fichiers d'en-tête C-appeler des prototypes, des types de données, des codes retour, des constantes et des structures</i>					
Nom de fichier	Description	IBM i	Systèmes AIX and Linux®	Windows	z/OS
<b>Appeler des prototypes, des types de données, des codes retour, des constantes et des structures</b>					
CMQC	Définitions MQI	C	C	C	C
CMQBC	Définitions MQAI	C	C	C	
CMQEC	Définition des points d'entrée d'interface (inclut CMQC, CMQXC et CMQZC)		C	C	
CMQCFC	Définitions PCF	C	C	C	C
CMQPSC	Définitions de publication / abonnement	C	C	C	C
CMQXC	Définitions de canal et de sortie	C	C	C	C
CMQZC	Définitions de services installables	C	C	C	
<b>Clé:</b> C= Fichiers fournis					

## Fichiers COBOL COPY

Divers fichiers COPY sont fournis pour vous aider à écrire des programmes d'application COBOL qui utilisent l'interface MQI.

Tableau 2. Fichiers de copie COBOL-codes retour, constantes et structures					
Nom de fichier	Description	IBM i	AIX and Linux	Windows	z/OS
<b>Codes retour et constantes</b>					
CMQx	Définitions MQI	V	V	V	V
CMQCFx	Définitions PCF	V	V	V	V
CMQPSx	Définitions de publication / abonnement	V	V	V	V
CMQXx	Définitions de canal et de sortie	V	V	V	V
<b>Structures</b>					
Récept. MQ	MQAIR-Enregistrement des informations d'authentification		V L	V L	
CMQBOx	MQBO-Options de début	V L	V L	V L	
CMQCDx	MQCD-Définition de canal	V L	V L	V L	V L
CMQCFBFx	MQCFBF-Paramètre de filtre de chaîne d'octets PCF	V L	V L	V L	V L
CMQCFBSx	MQCFBS-Paramètre de chaîne d'octets PCF	V L	V L	V L	V L
CMQCFGRx	MQCFGR-Paramètre de groupe PCF	V L	V L	V L	V L
CMQCFHx	En-tête MQCFH-PCF	V L	V L	V L	V L
CMQCFIFx	MQCFIF-Paramètre de filtre d'entier PCF	V L	V L	V L	V L
CMQCFILx	MQCFIL-Paramètre de liste d'entiers PCF	V L	V L	V L	V L
CMQCFINx	MQCFIN-Paramètre entier PCF	V L	V L	V L	V L
CMQCFSFx	MQCFSF-Paramètre de filtre de chaîne PCF	V L	V L	V L	V L
CMQCFSLx	MQCFSL-Paramètre de liste de chaînes PCF	V L	V L	V L	V L
CMQCFSTx	MQCFST-Paramètre de chaîne PCF	V L	V L	V L	V L
CMQCFXLx	MQCFIL64 -Paramètre de liste d'entiers PCF 64 bits	V L	V L	V L	V L
CMQCFXNx	MQCFIN64 -Paramètre d'entier PCF 64 bits	V L	V L	V L	V L
CMQCHRVx	MQCHARV-Chaîne de longueur variable	V L	V L	V L	V L
CMQCIHx	MQCIH-en-tête CICS bridge	V L	V L	V L	V L
CMQCNOx	MQCNO-Options de connexion	V L	V L	V L	V L

Tableau 2. Fichiers de copie COBOL-codes retour, constantes et structures (suite)

Nom de fichier	Description	IBM i	AIX and Linux	Windows	z/OS
CMQCSPx	MQCSP-Paramètres de sécurité	V L	V L	V L	V L
CMQCXPx	MQCXP-Paramètres d'exit de canal	V L			V L
CMQDHx	En-tête MQDH-Distribution	V L	V L	V L	V L
CMQDLHx	En-tête MQDLH-Dead-letter	V L	V L	V L	V L
CMQDXPx	MQDXP-Paramètres d'exit de conversion de données	V L		V L	
CMQEPHx	MQEPH-en-tête PCF imbriqué	V L	V L	V L	V L
CMQGMOx	MQGMO-Extraction des options de message	V L	V L	V L	V L
CMQIIHx	MQIIH - En-tête d'informations IMS	V L	V L	V L	V L
CMQMDx	MQMD-Descripteur de message	V L	V L	V L	V L
CMQMD1x	MQMD1 -Descripteur de message version 1	V L	V L	V L	V L
CMQMD2x	MQMD2 -Descripteur de message version 2	V L	V L	V L	V L
CMQMDEx	MQMDE-Descripteur de message étendu	V L	V L	V L	V L
CMQODx	MQOD-Descripteur d'objet	V L	V L	V L	V L
CMQORx	MQOR-Enregistrement d'objet	V L	V L	V L	V L
CMQPMOx	MQPMO-Options de message d'insertion	V L	V L	V L	V L
CMQRFHx	MQRFH-Règles et en-tête de formatage	V L	V L	V L	V L
CMQRFH2x	MQRFH2 - En-tête 2 de règles et de formatage	V L	V L	V L	V L
CMQRMHx	MQRMH-En-tête de message de référence	V L	V L	V L	V L
CMQRRx	MQRR-Enregistrement de réponse	V L	V L	V L	
CMQSCOx	MQSCO-Options de configuration TLS		V L	V L	
CMQTMx	MQTM-Message de déclenchement	V L		V L	V L
CMQTMcx	MQTMc-Caractère du message de déclenchement	V L	V L		
CMQTM2x	MQTM2 -Message de déclenchement 2 caractères	V L	V L	V L	V L
CMQWIHx	MQWIH-En-tête d'informations de travail	V L	V L	V L	V L
CMQXQHx	MQXQH-En-tête de file d'attente de transmission	V L	V L	V L	V L

Tableau 2. Fichiers de copie COBOL-codes retour, constantes et structures (suite)

Nom de fichier	Description	IBM i	AIX and Linux	Windows	z/OS
<b>Clé :</b>					
<ul style="list-style-type: none"> <li>Fichiers dont les valeurs initiales sont fournies, x = V</li> <li>Fichiers sans valeurs initiales fournies, x = L</li> </ul>					

### **z/OS Fichiers d'inclusion PL/I**

Un certain nombre de fichiers INCLUDE sont fournis pour le langage de programmation PL/I. Ces fichiers sont disponibles sous z/OS uniquement.

Tableau 3. Fichiers d'inclusion PL/I-types de données, codes retour, constantes et structures

Nom de fichier	Description	IBM i	AIX and Linux	Windows	z/OS
<b>Types de données, codes retour, constantes et structures</b>					
CMQP	Définitions MQI				P
CMQCFP	Définitions PCF				P
CMQEPP	Définitions de point d'entrée				P
CMQPSP	Définitions de publication / abonnement				P
CMQXP	Définitions de canal et de sortie				P
<b>Clé:</b> P= Fichier fourni					

### **IBM i Fichiers de copie RPG**

Les fichiers RPG COPY sont fournis pour le langage de programmation RPG. Ces fichiers sont disponibles uniquement sous IBM i.

Tableau 4. Fichiers de copie RPG-codes retour, constantes et structures

Nom de fichier	Description	IBM i	AIX and Linux	Windows	z/OS
<b>Codes retour et constantes</b>					
CMQx	Définitions MQI	G R			
CMQCFx	Définitions PCF	G			
CMQPSx	Définitions de publication / abonnement	G			
CMQXx	Définitions de canal et de sortie	G R			
<b>Structures</b>					
CMQBOX	MQBO-Options de début	G H			
CMQCDx	MQCD-Définition de canal	G H R			
CMQCFBFx	MQCFBF-Paramètre de filtre de chaîne d'octets PCF	G H			



Tableau 4. Fichiers de copie RPG-codes retour, constantes et structures (suite)

Nom de fichier	Description	IBM i	AIX and Linux	Windows	z/OS
CMQCFBSx	MQCFBS-Paramètre de chaîne d'octets PCF	G H			
CMQCFGRx	MQCFGR-Paramètre de groupe PCF	G H			
CMQCFHx	En-tête MQCFH-PCF	G H			
CMQCFIFx	MQCFIF-Paramètre de filtre d'entier PCF	G H			
CMQCFILx	MQCFIL-Paramètre de liste d'entiers PCF	G H			
CMQCFINx	MQCFIN-Paramètre entier PCF	G H			
CMQCFSFx	MQCFSF-Paramètre de filtre de chaîne PCF	G H			
CMQCFSLx	MQCFSL-Paramètre de liste de chaînes PCF	G H			
CMQCFSTx	MQCFST-Paramètre de chaîne PCF	G H			
CMQCFXLx	MQCFIL64 -Paramètre de liste d'entiers PCF 64 bits	G H			
CMQCFXNx	MQCFIN64 -Paramètre d'entier PCF 64 bits	G H			
CMQCHARVx	MQCHARV-Chaîne de longueur variable	G H			
CMQCIHx	MQCIH-en-tête CICS bridge	G H			
CMQCNOx	MQCNO-Options de connexion	G H			
CMQCSPx	MQCSP-Paramètres de sécurité	G H			
CMQCXPx	MQCXP-Paramètres d'exit de canal	G H R			
CMQDHx	En-tête MQDH-Distribution	G H R			
CMQDLHx	En-tête MQDLH-Dead-letter	G H R			
CMQDXPx	MQDXP-Paramètres d'exit de conversion de données	G H R			
CMQEPHx	MQEPH-en-tête PCF imbriqué	G H			
CMQGMOx	MQGMO-Extraction des options de message	G H R			
CMQIIHx	MQIIH - En-tête d'informations IMS	G H R			
CMQMDx	MQMD-Descripteur de message	G H R			
CMQMD1x	MQMD1 -Descripteur de message version 1	G H R			
CMQMD2x	MQMD2 -Descripteur de message version 2	G H			

Tableau 4. Fichiers de copie RPG-codes retour, constantes et structures (suite)

Nom de fichier	Description	IBM i	AIX and Linux	Windows	z/OS
CMQMDEx	MQMDE-Descripteur de message étendu	G H R			
CMQODx	MQOD-Descripteur d'objet	G H R			
CMQORx	MQOR-Enregistrement d'objet	G H R			
CMQPMOx	MQPMO-Options de message d'insertion	G H R			
CMQXPx	MQXP-Paramètres d'exit de routage de publication / abonnement	G H			
CMQRFHx	MQRFH-Règles et en-tête de formatage	G H			
CMQRFH2x	MQRFH2 - En-tête 2 de règles et de formatage	G H			
CMQRMHx	MQRMH-En-tête de message de référence	G H R			
CMQRRx	MQRR-Enregistrement de réponse	G H R			
CMQTMx	MQTM-Message de déclenchement	G H R			
CMQTMCx	MQTMC-Caractère du message de déclenchement	G H R			
CMQTM2x	MQTMC2 -Message de déclenchement 2 caractères	G H R			
CMQWIHx	MQWIH-En-tête d'informations de travail	G H			
CMQXQHx	MQXQH-En-tête de file d'attente de transmission	G H R			

**Clé :**

- Fichier pour liaison statique, initialisé, fourni x = G
- Fichier pour liaison statique, non initialisé, fourni x = H
- Fichier de liaison dynamique, initialisé, fourni, x = R

**Windows Fichiers du module Visual Basic**

Des fichiers d'en-tête (ou de formulaire) sont fournis pour vous aider à écrire des programmes d'application Visual Basic qui utilisent l'interface MQI. Ces fichiers d'en-tête sont fournis uniquement dans les versions 32 bits.

Tableau 5. Fichiers de module Visual Basic-déclarations d'appel, types de données, codes retour, constantes et structures

Nom de fichier	Description	IBM i	Systèmes AIX and Linux	Windows	z/OS
<b>Déclarations d'appel, types de données, codes retour, constantes et structures</b>					

Tableau 5. Fichiers de module Visual Basic-déclarations d'appel, types de données, codes retour, constantes et structures (suite)

Nom de fichier	Description	IBM i	Systèmes AIX and Linux	Windows	z/OS
CMQB	Définitions MQI			B	
CMQBB	Définitions MQAI			B	
CMQCFB	Définitions PCF			B	
CMQXB	Définitions de canal et de sortie			B	

**Clé:** B= Fichier fourni

### Fichiers COPY de l'assembleur z/OS

Divers fichiers COPY sont fournis pour vous aider à écrire des programmes d'application z/OS Assembler qui utilisent l'interface MQI.

Tableau 6. Fichiers de copie assembleur z/OS : types de données, codes retour, constantes et structures

Nom de fichier	Description	IBM i	AIX and Linux	Windows	z/OS
<b>Types de données, codes retour et constantes</b>					
CMQA	Définitions MQI				A
CMQCFA	Définitions PCF				A
CMQPSA	Définitions de publication / abonnement				A
CMQVERA	Contrôle de version de structure				A
CMQXA	Définitions de canal et de sortie				A
<b>Structures</b>					
CMQCDA	MQCD-Définition de canal				
CMQCFBFA	MQCFBF-Paramètre de filtre de chaîne d'octets PCF				
CMQCFBSA	MQCFBS-Paramètre de chaîne d'octets PCF				A
CMQCFGRA	MQCFGR-Paramètre de groupe PCF				A
CMQCFHA	En-tête MQCFH-PCF				A
CMQCFIFA	MQCFIF-Paramètre de filtre d'entier PCF				A
CMQCFILA	MQCFIL-Paramètre de liste d'entiers PCF				A
CMQCFINA	MQCFIN-Paramètre entier PCF				A
CMQCFSFA	MQCFSF-Paramètre de filtre de chaîne PCF				A
CMQCFSLA	MQCFSL-Paramètre de liste de chaînes PCF				A

Tableau 6. Fichiers de copie assembleur z/OS : types de données, codes retour, constantes et structures (suite)

Nom de fichier	Description	IBM i	AIX and Linux	Windows	z/OS
CMQCFSTA	MQCFST-Paramètre de chaîne PCF				A
CMQCFXLA	MQCFIL64 -Paramètre de liste d'entiers PCF 64 bits				A
CMQCFXNA	MQCFIN64 -Paramètre d'entier PCF 64 bits				A
CMQCHARVA	MQCHARV-Chaîne de longueur variable				A
CMQCIHA	MQCIH-en-tête CICS bridge				A
CMQCNOA	MQCNO-Options de connexion				A
CMQCSPA	MQCSP-Paramètres de sécurité				A
CMQCXPA	MQCXP-Paramètres d'exit de canal				A
CMQDHA	En-tête MQDH-Distribution				A
CMQDLHA	En-tête MQDLH-Dead-letter				A
CMQDXPA	MQDXP-Paramètres d'exit de conversion de données				A
CMQEPHA	MQEPH-en-tête PCF imbriqué				A
CMQGMOA	MQGMO-Extraction des options de message				A
CMQIIHA	MQIIH - En-tête d'informations IMS				A
CMQMDA	MQMD-Descripteur de message				A
CMQMD1A	MQMD1 -Descripteur de message version 1				A
CMQMD2A	MQMD2 -Descripteur de message version 2				A
CMQMDEA	MQMDE-Descripteur de message étendu				A
CMQODA	MQOD-Descripteur d'objet				A
CMQORA	MQOR-Enregistrement d'objet				A
CMQPMOA	MQPMO-Options de message d'insertion				A
CMQRFHA	MQRFH-Règles et en-tête de formatage				A
CMQRFH2A	MQRFH2 - En-tête 2 de règles et de formatage				A
CMQRMHA	MQRMH-En-tête de message de référence				A
Intervalle CMQTMA	MQTM-Message de déclenchement				A

Tableau 6. Fichiers de copie assembleur z/OS : types de données, codes retour, constantes et structures (suite)

Nom de fichier	Description	IBM i	AIX and Linux	Windows	z/OS
CMQTMCA	MQTMCA -Message de déclenchement 2 caractères				A
CMQWCRA	MQWCR-Enregistrement de cluster de charge de travail de cluster				A
CMQWDRA	MQWDR-Enregistrement de destination de charge de travail de cluster				A
CMQWDR1A	MQWDR1 -Enregistrement de destination de charge de travail de cluster version 1				A
CMQWDR2A	MQWDR2 -Enregistrement de destination de charge de travail de cluster version 2				A
CMQWIHA	MQWIH-En-tête d'informations de travail				A
CMQWQRA	MQWQR-Enregistrement de file d'attente de charge de travail de cluster				A
CMQWQR1A	MQWQR1 -Enregistrement de file d'attente de charge de travail de cluster version 1				A
CMQWQR2A	MQWQR2 -Enregistrement de file d'attente de charge de travail de cluster version 2				A
CMQWXP	MQWXP-Paramètres d'exit de charge de travail de cluster				A
CMQWXP1A	MQWXP1 -Paramètres d'exit de charge de travail de cluster version 1				A
CMQWXP2A	MQWXP2 -Paramètres d'exit de charge de travail de cluster version 2				A
CMQWXP3A	MQWXP3 -Paramètres d'exit de charge de travail de cluster version 3				A
CMQXPA	MQXP-Paramètres d'exit de croisement d'API CICS				A
CMQXQHA	MQXQH-En-tête de file d'attente de transmission				A
CMQXWDA	MQXWD-Descripteur d'attente d'exit				A

**Clé:** A= Fichier fourni

## MQ\_\* (longueurs de chaîne)

Tableau 7. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
LONGUEUR_CODE_ABEND_MQ_	4	X'00000004'
LONGUEUR_JETON_COMPTEUR_MQ	32	X'00000020'
LONGUEUR_NOM_FONCTION_APPL_MQ_	10	X'0000000A'
MQ_APPL_IDENTITY_DATA_LENGTH	32	X'00000020'
LONGUEUR_NOM_APPL_MQ_	28	X'0000001C'
MQ_APPL_ORIGINE_DATA_LENGTH	4	X'00000004'
LONGUEUR_BALISE_APPL_MQ_APPLICATION	28	X'0000001C'
LONGUEUR_SUFFIXE_ARM_MQ_	2	X'00000002'
LONGUEUR_ID_ATTENTION_MQ	4	X'00000004'
MQ_AUTH_INFO_NOM_CONN_LENGTH	264	X'00000108'
LONGUEUR_DESC_INFO_AUTH_MQ_MQ_AUTH_LENGTH	64	X'00000040'
LONGUEUR_NOM_INFO_AUTH_MQ_	48	X'00000030'
MQ_AUTH_INFO_OCSP_URL_LENGTH	256	X'00000100'
MQ_LONGUEUR_AUTHENTIFICATEUR	8	X'00000008'
MQ_AUTO_REORG_CATALOG_LENGTH	44	X'0000002C'
MQ_AUTO_REORG_TIME_LENGTH	4	X'00000004'
LONGUEUR_INTERFACE_ID_MQ_BATCH_	8	X'00000008'
LONGUEUR_NOM_PONT	24	X'00000018'
LONGUEUR_CODE_ANNULATION	4	X'00000004'
MQ_CF_STRUC_DESC_LENGTH	64	X'00000040'
LONGUEUR_NOM_STRUC_MQ_CF_	12	X'0000000C'
LONGUEUR_DATE_CANAL_MQ_CANAL	12	X'0000000C'
LONGUEUR_CANAL_MQ_CANAL	64	X'00000040'
LONGUEUR_NOM_CANAL_MQ_CANAL	20	X'00000014'
LONGUEUR_TEMPS_CANAL_MQ_CANAL	8	X'00000008'
LONGUEUR_PARAMÈTRE_SERVICE_MQ_CHINIT_SERVICE	32	X'00000020'
LONGUEUR_NOM_FICHIERS_MQ_CICS	8	X'00000008'
LONGUEUR_ID_CLIENT_MQ	23	X'00000017'
LONGUEUR_NOM_CLUSTER	48	X'00000030'
LONGUEUR_NOM_CONN_MQ_	264	X'00000108'
LONGUEUR_CONN_MQ_DE_CONNEXION	128	X'00000080'
LONGUEUR_ID_CONNEXION_MQM	24	X'00000018'
LONGUEUR_ID_CORREL_MQ_	24	X'00000018'
LONGUEUR_DATE_CREATION_MQ_	12	X'0000000C'
LONGUEUR_DURÉE_CREATION_MQ_	8	X'00000008'
LONGUEUR_DATE_MQ	12	X'0000000C'
LONGUEUR_NOM_DISTINCTIF	1024	X'00000400'
MQ_DNS_GROUP_NAME_LENGTH	18	X'00000012'

Tableau 7. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQ_LONGUEUR_DONNÉES_SORTIE	32	X'00000020'
LONGUEUR_NOM_INFO_SORTIE	48	X'00000030'
LONGUEUR_NOM_SORTIE	(value differs by platform or version)	
MQ_EXIT_PD_LONGUEUR_ZONE	48	X'00000030'
MQ_EXIT_USER_LONGUEUR_ZONE	16	X'00000010'
LONGUEUR_FACILIT_MQM	8	X'00000008'
LONGUEUR_FACILIT_MQ_LIKE_LENGTH	4	X'00000004'
LONGUEUR_FORMAT_MQ	8	X'00000008'
LONGUEUR_FONCTION_MQS	4	X'00000004'
LONGUEUR_GROUPE_MQ_ID	24	X'00000018'
LONGUEUR_MOT_DE_PASSE_LDAP_MQ_	32	X'00000020'
LONGUEUR_NOM_LISTE_MQ_ÉCOUTE	48	X'00000030'
LONGUEUR_DESC_LISTE_MQ_ÉCOUTE	64	X'00000040'
LONGUEUR_ADRESSE_LOCAL_MQ_	48	X'00000030'
MQ_LTERM_OVERRIDE_LENGTH	8	X'00000008'
LONGUEUR_NOM_MQ_LU_LENGTH	8	X'00000008'
LONGUEUR_LARGEUR_MQ_LUM	16	X'00000010'
LONGUEUR_NOM_EXIT_MQ_MAX	128	X'00000080'
MQ_MAX_MCA_LONGUEUR_ID_UTILISATEUR	64	X'00000040'
LONGUEUR_NOM_PROPRIÉTÉ_MAX_MQ_MAX	4095	X'00000FFF'
MQ_MAX_LONGUEUR_ID_UTILISATEUR	64	X'00000040'
LONGUEUR_NOM_TRAVAIL_MQ_MCA_MCA_	28	X'0000001C'
LONGUEUR_NOM_MQ_MCA_	20	X'00000014'
MQ_MCA_LONGUEUR_DONNÉES_UTILISATEUR	32	X'00000020'
MQ_MCA_LONGUEUR_ID_UTILISATEUR	(value differs by platform or version)	
MQ_MFS_MAP_NAME_LENGTH	8	X'00000008'
LONGUEUR_NOM_MODE_MQ_	8	X'00000008'
MQ_MSG_HEADER_LENGTH	4000	X'00000FA0'
LONGUEUR_MSG_MQ_ID	24	X'00000018'
LONGUEUR_JETON_MSG	16	X'00000010'
LONGUEUR_DESC_NOM_MQ_NAME	64	X'00000040'
LONGUEUR_NOM_MQ_NOM	48	X'00000030'
LONGUEUR_NOM_INSTANCE_MQ_NHA_	48	X'00000030'
LONGUEUR_INSTANCE_OBJET_MQ_ID	24	X'00000018'
LONGUEUR_NOM_OBJET_MQ_	48	X'00000030'
MQ_PASS_TICKET_APPL_LENGTH	8	X'00000008'
LONGUEUR_MOT_DE_PASSE_MQM	12	X'0000000C'
LONGUEUR_ID_APP_PROCESSUS_MQM	256	X'00000100'
LONGUEUR_DES_PROCESSUS_MQ_TRAITEMENT	64	X'00000040'

Tableau 7. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQ_PROCESS_ENV_DATA_LENGTH	128	X'00000080'
LONGUEUR_NOM_PROCESSUS	48	X'00000030'
LONGUEUR_DONNÉES_UTILISATEUR_PROCESSUS_MQ_UTILISATEUR	128	X'00000080'
LONGUEUR_NOM_PROGRAMME	20	X'00000014'
MQ_PUT_APPL_NAME_LENGTH	28	X'0000001C'
LONGUEUR_DATE_PUT_MQ_	8	X'00000008'
LONGUEUR_TEMPS_PUT_MQ_LONG	8	X'00000008'
LONGUEUR_DESC_Q_MQ_LONG	64	X'00000040'
MQ_Q_MGR_DESC_LENGTH	64	X'00000040'
MQ_Q_MGR_IDENTIFIÉ_LENGTH	48	X'00000030'
MQ_Q_MGR_NAME_LENGTH	48	X'00000030'
LONGUEUR_NOM_Q_MQ_	48	X'00000030'
LONGUEUR_NOM_QSG_MQ_	4	X'00000004'
LONGUEUR_SYS_DISTANT_MQ_ID	4	X'00000004'
LONGUEUR_ID_SÉCURITÉ_MQ	40	X'00000028'
LONGUEUR_SÉLECTEUR_MQM	10240	X'00002800'
LONGUEUR_ARGS_SERVICE_MQ_LONG	255	X'000000FF'
LONGUEUR_COMMANDE_SERVICE_MQ_SERVICE	255	X'000000FF'
LONGUEUR_DESC_SERVICE_MQ_SERVICE	64	X'00000040'
LONGUEUR_NOM_SERVICE_MQ_	32	X'00000020'
LONGUEUR_CHEMIN_SERVICE_MQ_LONG	255	X'000000FF'
LONGUEUR_ÉTAPE_SERVICE_MQ_SERVICE	8	X'00000008'
MQ_SHORT_CONN_NAME_LENGTH	20	X'00000014'
MQ_SHORT_DNAME_LENGTH	256	X'00000100'
MQ_SSL_CIPHER_SPEC_LENGTH	32	X'00000020'
MQ_SSL_CRYPTO_LONGUEUR_MATÉRIEL	256	X'00000100'
MQ_SSL_HANDSHAKE_STAGE_LENGTH	32	X'00000020'
LONGUEUR_BIBLIOTHÈQUE_SSL_MQ_SSL_CLÉ	44	X'0000002C'
MQ_SSL_KEY_MEMBER_LENGTH	8	X'00000008'
LONGUEUR RÉFÉRENTIEL_CLÉ_SSL_MQ_SSL	256	X'00000100'
MQ_SSL_PEER_NAME_LENGTH	1024	X'00000400'
MQ_SSL_SHORT_PEER_NAME_LENGTH	256	X'00000100'
LONGUEUR_CODE_DÉMARRAGE	4	X'00000004'
LONGUEUR_CLASSE_STOCKAGE_MQ_DESC_LENGTH	64	X'00000040'
LONGUEUR_CLASSE_STOCKAGE_MQM	8	X'00000008'
MQ_SUB_IDENTITY_LENGTH	128	X'00000080'
LONGUEUR_POINT_SOUS-FILE	128	X'00000080'
MQ_SUITE_B_128_BIT	2	X'00000002'
MQ_SUITE_B_192_BIT	4	X'00000004'



<i>Tableau 7. Valeurs des constantes (suite)</i>		
<b>Nom</b>	<b>Valeur décimale</b>	<b>Valeur hexadécimale</b>
MQ_SUITE_B_NONE	1	X'00000001'
MQ_SUITE_B_NOT_AVAILABLE	0	X'00000000'
MQ_TCP_NAME_LENGTH	8	X'00000008'
LONGUEUR_TEMPS_MQ_ATTENTE	8	X'00000008'
LONGUEUR_DESC_TOPIC_MQ_LONG	64	X'00000040'
LONGUEUR_NOM_TOPIC_MQ_	48	X'00000030'
LONGUEUR_TOPIC_MQ_STR_LENGTH	10240	X'00002800'
LONGUEUR_EXIT_TOTAL_MQ_DONNÉES	999	X'000003E7'
LONGUEUR_NOM_EXIT_TOTAL_MQ_	999	X'000003E7'
LONGUEUR_NOM_MQ_TP_LENGTH	64	X'00000040'
MQ_TPIPE_NAME_LENGTH	8	X'00000008'
LONGUEUR_ID_INSTANCE_MQ_TRANS	16	X'00000010'
LONGUEUR_ID_TRANSACTION_MQM	4	X'00000004'
LONGUEUR_DONNÉES_TRIGGER_MQ_	64	X'00000040'
LONGUEUR_NOM_PROGRAMME_TRIGGER_MQ_	8	X'00000008'
MQ_TRIGGER_TERM_ID_LENGTH	4	X'00000004'
MQ_TRIGGER_TRANS_ID_LENGTH	4	X'00000004'
LONGUEUR_ID_UTILISATEUR	12	X'0000000C'
LONGUEUR_VERSION_MQ	8	X'00000008'
LONGUEUR_NOM_GROUPE_XCF_MQ_XCF	8	X'00000008'
MQ_XCF_NOM_MEMBRE_LENGTH	16	X'00000010'

### **MQ\_ \* (Longueur de chaîne de format de commande)**

<i>Tableau 8. Valeurs des constantes</i>		
<b>Nom</b>	<b>Valeur décimale</b>	<b>Valeur hexadécimale</b>
MQ_ARCHIVE_PFX_LENGTH	36	X'00000024'
LONGUEUR_UNITÉ_ARCHIVE_MQ_M	8	X'00000008'
MQ_ASID_LENGTH	4	X'00000004'
LONGUEUR_NOM_PROFIL_AUTH_MQ_	48	X'00000030'
LONGUEUR_ID_MQ_CF_LEID	12	X'0000000C'
LONGUEUR_MQ_COMMANDE_COMMANDE	32768	X'00008000'
LONGUEUR_NOM_JEU_DONNÉES_MQ_DONNÉES	44	X'0000002C'
MQ_DB2_NAME_LENGTH	4	X'00000004'
LONGUEUR_NOM_DSG_MQ_	8	X'00000008'
LONGUEUR_NOM_ENTITÉ	1024	X'00000400'
MQ_ENV_INFO_LENGTH	96	X'00000060'
LONGUEUR_ADRESSE_IP_MQ_	48	X'00000030'
LONGUEUR_ID_CORREL_LOG_MQ_	8	X'00000008'
LONGUEUR_NOM_EXT_JOURNAL_MQ_LOG	24	X'00000018'
LONGUEUR_CHEMIN_JOURNAL_MQ	1024	X'00000400'

Tableau 8. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQ_LRSN_LENGTH	12	X'0000000C'
LONGUEUR_NOM_ORIGINE_MQ_	8	X'00000008'
LONGUEUR_NOM_MQ_PSB	8	X'00000008'
LONGUEUR_ID_INSTRUCTION	8	X'00000008'
MQ_Q_MGR_CPF_LENGTH	4	X'00000004'
LONGUEUR_RÉPONSE_MQ_ID	24	X'00000018'
MQ_RBA_LENGTH	16	X'00000010'
LONGUEUR_PROFIL_SÉCURITÉ_MQM	40	X'00000028'
LONGUEUR_COMPOSANT_SERVICE_MQ_SERVICE	48	X'00000030'
LONGUEUR_NOM_SOUS-FILE	10240	X'00002800'
LONGUEUR_SERVICE_MQ_SYSP_SERVICE	32	X'00000020'
LONGUEUR_NOM_SYSTÈME	8	X'00000008'
NOMBRE_TÂCHE_MQ_LONGUEUR	8	X'00000008'
MQ_TPIPE_PFX_LENGTH	4	X'00000004'
LONGUEUR_ID_UOW_MQ_	256	X'00000100'
LONGUEUR_DONNÉES_UTILISATEUR_MQ	10240	X'00002800'
LONGUEUR_VOL_MQM	6	X'00000006'

### MQACH\_\* (structure d'en-tête de zone de chaîne d'exit d'API)

Tableau 9. Structures de constantes	
Nom	Structure
ID_STRUC_MQACH_	"ACH↵"
MQACH_STRUC_ID_ARRAY	'A','C','H','↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 10. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQACH_VERSION_1	1	X'00000001'
VERSION_MQACH_CURRENT_VERSION	1	X'00000001'
MQACH_LENGTH_1	(value differs by platform or version)	(value differs by platform or version)
MQACH_LONGUEUR_ACTUELLE	(value differs by platform or version)	(value differs by platform or version)

### MQACT\_\* (jeton de comptabilité)

Tableau 11. Noms et valeurs de constante	
Nom	Valeur
MQACT_AUCUN	X'00...00' (32 valeurs nulles)
MQACT_NON_ARRAY	'\0','\0',... (32 valeurs nulles)

## MQACT\_\* (Options d'action de format de commande)

Tableau 12. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQACT_FORCE_REMOVE	1	X'00000001'
MQACT_ADVANCE_LOG	2	X'00000002'
MQACT_COLLECT_STATISTICS	3	X'00000003'
MQACT_PUBSUB	4	X'00000004'

## MQACTP\_\* (Action)

Tableau 13. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQACTP_NOUVEAU	0	X'00000000'
MQACTP_FORWARD	1	X'00000001'
MQACTP_REPLY	2	X'00000002'
RAPPORT MQACTP_RAPPORT	3	X'00000003'

## MQACTT\_\* (types de jeton de comptabilité)

Tableau 14. Valeurs des constantes	
Nom	Valeur hexadécimale
MQACTT_INCONNU	X'00'
MQACTT_CICS_LUOW_ID	X'01'
MQACTT_OS2_DEFAULT	X'04'
MQACTT_DOS_DEFAULT	X'05'
MQACTT_UNIX_NUMERIC_ID	X'06'
MQACTT_OS400_ACCOUNT_TOKEN	X'08'
MQACTT_WINDOWS_DEFAULT	X'09'
MQACTT_NT_SECURITY_ID	X'0B'
MQACTT_UTILISATEUR	X'19'

## MQADOPT\_\* (adoption de nouveaux contrôles MCA et adoption de nouveaux types MCA)

### Adopter les nouvelles vérifications MCA

Tableau 15. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQADOPT_CHECK_NONE	0	X'00000000'
MQADOPT_CHECK_ALL	1	X'00000001'
MQADOPT_CHECK_Q_MGR_NAME	2	X'00000002'
MQADOPT_CHECK_NET_ADDR	4	X'00000004'

## Adopter de nouveaux types MCA

Tableau 16. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQADOPT_TYPE_NO	0	X'00000000'
MQADOPT_TYPE_ALL	1	X'00000001'
MQADOPT_TYPE_SVR	2	X'00000002'
MQADOPT_TYPE_SDR	4	X'00000004'
MQADOPT_TYPE_RCVR	8	X'00000008'
MQADOPT_TYPE_CLUSRCVR	16	X'00000010'

## MQAIR\_\* (Structure d'enregistrement des informations d'authentification)

Tableau 17. Structures de constantes

Nom	Structure
ID_STRUC_MQAIR_	"AIR↵"
MQAIR_STRUC_ID_ARRAY	'A', 'I', 'R', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 18. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQAIR_VERSION_1	1	X'00000001'
MQAIR_VERSION_2	2	X'00000002'
MQAIR_CURRENT_VERSION	2	X'00000002'

## MQAIT\_\* (type d'informations d'authentification)

Tableau 19. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQAIT_ALL	0	X'00000000'
MQAIT_CRL_LDAP	1	X'00000001'
MQAIT_OCSP	2	X'00000002'
MQAIT_IDPW_OS	3	X'00000003'
MQAIT_IDPW_LDAP	4	X'00000004'

## MQAS\_\* (Valeurs d'état asynchrones du format de commande)

Tableau 20. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQAS_AUCUN	0	X'00000000'
MQAS_DEMARRE	1	X'00000001'
MQAS_START_WAIT	2	X'00000002'
MQAS_ARRETEM	3	X'00000003'
MQAS_SUSPENDU	4	X'00000004'
MQAS_SUSPENDU_TEMPORAIRE	5	X'00000005'
MQAS_ACTIF	6	X'00000006'

Tableau 20. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQAS_XX_ENCODE_CASE_ONE inactif	7	X'00000007'

### MQAT\_\* (types d'application d'insertion)

Tableau 21. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQAT_INCONNU	-1	X'FFFFFFFF'
MQAT_NO_CONTEXT	0	X'00000000'
MQAT_CICS	1	X'00000001'
MQAT_MVS	2	X'00000002'
MQAT_OS390	2	X'00000002'
MQAT_ZOS	2	X'00000002'
MQAT_IMS	3	X'00000003'
MQAT_OS2	4	X'00000004'
MQAT_DOS	5	X'00000005'
MQAT_AIX	6	X'00000006'
MQAT_UNIX	6	X'00000006'
MQAT_QMGR	7	X'00000007'
MQAT_OS400	8	X'00000008'
MQAT_WINDOWS	9	X'00000009'
MQAT_CICS_VSE	10	X'0000000A'
MQAT_WINDOWS_NT	11	X'0000000B'
MQAT_VMS	12	X'0000000C'
MQAT_GARDIEN	13	X'0000000D'
MQAT_NSK	13	X'0000000D'
MQAT_VOS	14	X'0000000E'
MQAT_OPEN_TP1	15	X'0000000F'
MQAT_MV	18	X'00000012'
MQAT_IMS_BRIDGE	19	X'00000013'
MQAT_XCF	20	X'00000014'
MQAT_CICS_BRIDGE	21	X'00000015'
MQAT_NOTES_AGENT	22	X'00000016'
MQAT_TPF	23	X'00000017'
Utilisateur_MQAT	25	X'00000019'
MQAT_COURTIER	26	X'0000001A'
MQAT_QMGR_PUBLISH	26	X'0000001A'
MQAT_JAVA	28	X'0000001C'
MQAT_DQM	29	X'0000001D'
MQAT_CHANNEL_INITIATOR	30	X'0000001E'
MQAT_WLM	31	X'0000001F'

Tableau 21. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQAT_BATCH	32	X'00000020'
MQAT_RRS_BATCH	33	X'00000021'
MQAT_SIB	34	X'00000022'
MQAT_PAR_DEFAULT	(value differs by platform or version)	(value differs by platform or version)
MQAT_USER_FIRST	65536	X'00010000'
MQAT_USER_LAST	99999999	X'3B9AC9FF'

### MQAUTH\_\* (Valeurs des droits de format de commande)

Tableau 22. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQAUTH_NONE	0	X'00000000'
MQAUTH_ALT_USER_AUTHORITY	1	X'00000001'
MQAUTH_BROWSE	2	X'00000002'
CHANGE_MQAUTH_CHANGE	3	X'00000003'
MQAUTH_CLEAR	4	X'00000004'
MQAUTH_CONNECT	5	X'00000005'
CREER MQAUTH_CREATE	6	X'00000006'
MQAUTH_DELETE	7	X'00000007'
MQAUTH_DISPLAY	8	X'00000008'
MQAUTH_INPUT	9	X'00000009'
MQAUTH_INQUIRE	10	X'0000000A'
MQAUTH_OUTPUT	11	X'0000000B'
MQAUTH_PASS_ALL_CONTEXT	12	X'0000000C'
MQAUTH_PASS_IDENTITY_CONTEXT	13	X'0000000D'
SET MQAUTH_SET	14	X'0000000E'
MQAUTH_SET_ALL_CONTEXT	15	X'0000000F'
MQAUTH_SET_IDENTITY_CONTEXT	16	X'00000010'
CONTROLE MQAUTH_DE CONTROLE	17	X'00000011'
MQAUTH_CONTRÔLE_ÉTENDU	18	X'00000012'
MQAUTH_PUBLICATION	19	X'00000013'
MQAUTH_ABONNEMENT	20	X'00000014'
MQAUTH_RESUME	21	X'00000015'
SYSTEME MQAUTH_	22	X'00000016'

### MQAUTHOPT\_\* (Options des droits de format de commande)

Tableau 23. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQAUTHOPT_CUMULATIVE	256	X'00000100'
MQAUTHOPT_ENTITY_EXPLICIT	1	X'00000001'

Tableau 23. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQAUTHOPT_ENTITY_SET	2	X'00000002'
MQAUTHOPT_NOM_ALL_MATCHING	32	X'00000020'
MQAUTHOPT_NAME_AS_WILDCARD	64	X'00000040'
MQAUTHOPT_NOM_EXPLICITE	16	X'00000010'

### MQAXC\_\* (structure de contexte d'exit d'API)

Tableau 24. Structures de constantes	
Nom	Structure
ID_STRUC_MQAXC	"AXC↵"
MQAXC_STRUC_ID_ARRAY	'A', 'X', 'C', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 25. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQAXC_VERSION_1	1	X'00000001'
MQAXC_CURRENT_VERSION	1	X'00000001'

### MQAXP\_\* (structure des paramètres d'exit d'API)

Tableau 26. Structures de constantes	
Nom	Structure
ID_STRUC_MQAXP	"AXP↵"
MQAXP_STRUC_ID_ARRAY	'A', 'X', 'P', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 27. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQAXP_VERSION_1	1	X'00000001'
MQAXP_VERSION_2	2	X'00000002'
MQAXP_CURRENT_VERSION	2	X'00000002'

### MQBA\_\* (Sélecteurs d'attribut d'octet)

Tableau 28. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQBA_FIRST	6001	X'00001771'
MQBA_LAST	8000	X'00001F40'

### MQBACF\_\* (types de paramètre d'octet de format de commande)

Tableau 29. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQBACF_PREMIER	7001	X'00001B59'

Tableau 29. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQBACF_EVENT_ACCOUNTING_TOKEN	7001	X'00001B59'
ID_SÉCURITÉ_ÉVÉNEMENT_MQBACF	7002	X'00001B5A'
MQBACF_RESPONSE_SET	7003	X'00001B5B'
ID_RÉPONSE_MQBACF	7004	X'00001B5C'
ID_UOW_EXTERNAL_MQBACF	7005	X'00001B5D'
ID_CONNEXION_MQBACF	7006	X'00001B5E'
ID_CONNEXION_GENERIC_MQBACF	7007	X'00001B5F'
MQBACF_ORIGIN_UOW_ID	7008	X'00001B60'
MQBACF_Q_MGR_UOW_ID	7009	X'00001B61'
JETON_COMPTEUR_MQBACF	7010	X'00001B62'
ID_CORREL_MQBACF	7011	X'00001B63'
ID_GROUPE_MQBACF	7012	X'00001B64'
ID_MSG_MQBACF	7013	X'00001B65'
MQBACF_CF_LEID	7014	X'00001B66'
ID_CORREL_DESTINATION_MQBACF	7015	X'00001B67'
ID_SUB_MQBACF	7016	X'00001B68'
MQBACF_LAST_UTILISÉ	7016	X'00001B68'

### MQBL\_\* (Longueur de la mémoire tampon pour la chaîne mqAddet la chaîne mqSet)

Tableau 30. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQBL_NULL_CLOS	-1	X'FFFFFFFF'

### MQBMHO\_\* (options et structure de la mémoire tampon pour le traitement des messages)

#### Structure des options de gestion de la mémoire tampon vers le message

Tableau 31. Structures de constantes	
Nom	Structure
ID_STRUC_MQBMHO_	"BMHO"
MQBMHO_STRUC_ID_ARRAY	'B', 'M', 'H', 'O'

**Remarque :** Le symbole – représente un caractère blanc unique.

Tableau 32. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQBMHO_VERSION_1	1	X'00000001'
MQBMHO_VERSION	1	X'00000001'



## Options de traitement de la mémoire tampon vers le message

Tableau 33. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQBMHO_AUCUN	0	X'00000000'
MQBMHO_DELETE_PROPERTIES	1	X'00000001'

## MQBND\_\* (liaisons par défaut)

Tableau 34. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQBND_BIND_ON_OPEN	0	X'00000000'
MQBND_BIND_NOT_FIXED	1	X'00000001'
MQBND_BIND_ON_GROUPE	2	X'00000002'

## MQBO\_\* (Options de début et structure)

### Structure des options de début

Tableau 35. Structures de constantes	
Nom	Structure
ID_STRUC_MQBO_	"B0¬¬"
MQBO_STRUC_ID_ARRAY	'B', '0', '¬', '¬'

**Remarque :** Le symbole ¬ représente un caractère blanc unique.

Tableau 36. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQBO_VERSION_1	1	X'00000001'
MQBO_CURRENT_VERSION	1	X'00000001'

## Options de début

Tableau 37. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQBO_AUCUN	0	X'00000000'

## MQBT\_\* (types de pont de format de commande)

Tableau 38. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQBT_OTMA	1	X'00000001'

## MQCA\_\* (Sélecteurs d'attribut de caractère)

Tableau 39. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCA_ADMIN_TOPIC_NAME	2105	X'00000839'
MQCA_ALTERATION_DATE	2027	X'000007EB'

Tableau 39. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQCA_ALTERATION_TIME	2028	X'000007EC'
MQCA_APPL_ID	2001	X'000007D1'
MQCA_AUTH_INFO_CONN_NAME	2053	X'00000805'
MQCA_AUTH_INFO_DESC	2046	X'000007FE'
MQCA_AUTH_INFO_NAME	2045	X'000007FD'
MQCA_AUTH_INFO_OCSP_URL	2109	X'0000083D'
MQCA_AUTO_REORG_CATALOG	2091	X'0000082B'
MQCA_AUTO_REORG_START_TIME	2090	X'0000082A'
MQCA_BACKOUT_REQ_Q_NAME	2019	X'000007E3'
MQCA_BASE_OBJECT_NAME	2002	X'000007D2'
MQCA_BASE_Q_NAME	2002	X'000007D2'
MQCA_BATCH_INTERFACE_ID	2068	X'00000814'
MQCA_CF_STRUC_DESC	2052	X'00000804'
MQCA_CF_STRUC_NAME	2039	X'000007F7'
MQCA_CHANNEL_AUTO_DEF_EXIT	2026	X'000007EA'
MQCA_CHILD	2101	X'00000835'
MQCA_CHINIT_SERVICE_PARM	2076	X'0000081C'
MQCA_CICS_FILE_NAME	2060	X'0000080C'
MQCA_CLUS_CHL_NAME	2124	X'0000084C'
MQCA_CLUSTER_DATE	2037	X'000007F5'
MQCA_CLUSTER_NAME	2029	X'000007ED'
MQCA_CLUSTER_NAMELIST	2030	X'000007EE'
MQCA_CLUSTER_Q_MGR_NAME	2031	X'000007EF'
MQCA_CLUSTER_TIME	2038	X'000007F6'
MQCA_CLUSTER_WORKLOAD_DATA	2034	X'000007F2'
MQCA_CLUSTER_WORKLOAD_EXIT	2033	X'000007F1'
MQCA_COMMAND_INPUT_Q_NAME	2003	X'000007D3'
MQCA_COMMAND_REPLY_Q_NAME	2067	X'00000813'
MQCA_CREATION_DATE	2004	X'000007D4'
MQCA_CREATION_TIME	2005	X'000007D5'
MQCA_DEAD_LETTER_Q_NAME	2006	X'000007D6'
MQCA_DEF_XMIT_Q_NAME	2025	X'000007E9'
MQCA_DNS_GROUP	2071	X'00000817'
MQCA_ENV_DATA	2007	X'000007D7'
MQCA_FIRST	2001	X'000007D1'
MQCA_IGQ_USER_ID	2041	X'000007F9'
MQCA_INITIATION_Q_NAME	2008	X'000007D8'
MQCA_LAST	4000	X'00000FA0'
MQCA_LAST_USED	2109	X'0000083D'

Tableau 39. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQCA_LDAP_PASSWORD	2048	X'00000800'
MQCA_LDAP_USER_NAME	2047	X'000007FF'
MQCA_LU_GROUP_NAME	2072	X'00000818'
MQCA_LU_NAME	2073	X'00000819'
MQCA_LU62_ARM_SUFFIX	2074	X'0000081A'
MQCA_MODEL_DURABLE_Q	2096	X'00000830'
MQCA_MODEL_NON_DURABLE_Q	2097	X'00000831'
MQCA_MONITOR_Q_NAME	2066	X'00000812'
MQCA_NAMELIST_DESC	2009	X'000007D9'
MQCA_NAMELIST_NAME	2010	X'000007DA'
MQCA_NAMES	2020	X'000007E4'
MQCA_PARENT	2102	X'00000836'
MQCA_PASS_TICKET_APPL	2086	X'00000826'
MQCA_PROCESS_DESC	2011	X'000007DB'
MQCA_PROCESS_NAME	2012	X'000007DC'
MQCA_Q_DESC	2013	X'000007DD'
MQCA_Q_MGR_DESC	2014	X'000007DE'
MQCA_Q_MGR_IDENTIFIER	2032	X'000007F0'
MQCA_Q_MGR_NAME	2015	X'000007DF'
MQCA_Q_NAME	2016	X'000007E0'
MQCA_QSG_NAME	2040	X'000007F8'
MQCA_REMOTE_Q_MGR_NAME	2017	X'000007E1'
MQCA_REMOTE_Q_NAME	2018	X'000007E2'
MQCA_REPOSITORY_NAME	2035	X'000007F3'
MQCA_REPOSITORY_NAMELIST	2036	X'000007F4'
MQCA_RESUME_DATE	2098	X'00000832'
MQCA_RESUME_TIME	2099	X'00000833'
MQCA_SERVICE_DESC	2078	X'0000081E'
MQCA_SERVICE_NAME	2077	X'0000081D'
MQCA_SERVICE_START_ARGS	2080	X'00000820'
MQCA_SERVICE_START_COMMAND	2079	X'0000081F'
MQCA_SERVICE_STOP_ARGS	2082	X'00000822'
MQCA_SERVICE_STOP_COMMAND	2081	X'00000821'
MQCA_STDERR_DESTINATION	2084	X'00000824'
MQCA_STDOUT_DESTINATION	2083	X'00000823'
MQCA_SSL_CRL_NAMELIST	2050	X'00000802'
MQCA_SSL_CRYPTO_HARDWARE	2051	X'00000803'
MQCA_SSL_KEY_LIBRARY	2069	X'00000815'
MQCA_SSL_KEY_MEMBER	2070	X'00000816'

Tableau 39. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQCA_SSL_KEY_REPOSITORY	2049	X'00000801'
MQCA_STORAGE_CLASS	2022	X'000007E6'
MQCA_STORAGE_CLASS_DESC	2042	X'000007FA'
MQCA_SYSTEM_LOG_Q_NAME	2065	X'00000811'
MQCA_TCP_NAME	2075	X'0000081B'
MQCA_TOPIC_DESC	2093	X'0000082D'
MQCA_TOPIC_NAME	2092	X'0000082C'
MQCA_TOPIC_STRING_FILTER	2108	X'0000083C'
MQCA_TOPIC_STRING	2094	X'0000082E'
MQCA_TPIPE_NAME	2085	X'00000825'
MQCA_TRIGGER_CHANNEL_NAME	2064	X'00000810'
MQCA_TRIGGER_DATA	2023	X'000007E7'
MQCA_TRIGGER_PROGRAM_NAME	2062	X'0000080E'
MQCA_TRIGGER_TERM_ID	2063	X'0000080F'
MQCA_TRIGGER_TRANS_ID	2061	X'0000080D'
MQCA_USER_DATA	2021	X'000007E5'
MQCA_USER_LIST	4000	X'00000FA0'
MQCA_VERSION	2120	X'00000848'
MQCA_XCF_GROUP_NAME	2043	X'000007FB'
MQCA_XCF_MEMBER_NAME	2044	X'000007FC'
MQCA_XMIT_Q_NAME	2024	X'000007E8'

### MQCACF\_\* (Types de paramètre de caractère de format de commande)

Tableau 40. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCACF_FIRST	3001	X'00000BB9'
MQCACF_NOM_Q_SOURCE	3001	X'00000BB9'
MQCACF_TO_NOM_FILE	3002	X'00000BBA'
MQCACF_NOM_PROCESSUS_SOURCE	3003	X'00000BBB'
MQCACF_TO_NOM_PROCESSUS	3004	X'00000BBC'
MQCACF_NOM_NAMELIST_SOURCE	3005	X'00000BBD'
MQCACF_TO_NOM_LISTE	3006	X'00000BBE'
MQCACF_NOM_CANAL_SOURCE	3007	X'00000BBF'
MQCACF_TO_CHANNEL_NAME	3008	X'00000BC0'
MQCACF_NOM_INFO_AUTH_SOURCE	3009	X'00000BC1'
MQCACF_TO_AUTH_INFO_NAME	3010	X'00000BC2'
MQCACF_Q_NOMS	3011	X'00000BC3'
NOMS_PROCESSUS_MQCACF	3012	X'00000BC4'
MQCACF_NAMELIST_NAMES	3013	X'00000BC5'

Tableau 40. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
TEXTE_ÉCHAPPATION_MQCACF	3014	X'00000BC6'
MQCACF_LOCAL_Q_NAMES	3015	X'00000BC7'
MQCACF_MODEL_Q_NAMES	3016	X'00000BC8'
MQCACF_ALIAS_Q_NAMES	3017	X'00000BC9'
MQCACF_REMOTE_Q_NAMES	3018	X'00000BCA'
MQCACF_SENDER_CHANNEL_NAMES	3019	X'00000BCB'
MQCACF_SERVER_CHANNEL_NAMES	3020	X'00000BCC'
MQCACF_REQUESTER_CHANNEL_NAMES	3021	X'00000BCD'
MQCACF_NOM_CANAL_RÉCEPTEUR	3022	X'00000BCE'
MQCACF_OBJECT_Q_MGR_NAME	3023	X'00000BCF'
NOM_APPL_MQCACF	3024	X'00000BD0'
MQCACF_IDENTIFICATEUR_UTILISATEUR	3025	X'00000BD1'
MQCACF_AUX_ERROR_DATA_STR_1	3026	X'00000BD2'
MQCACF_AUX_ERROR_DATA_STR_2	3027	X'00000BD3'
MQCACF_AUX_ERROR_DATA_STR_3	3028	X'00000BD4'
NOM_BRIDGE_MQCACF	3029	X'00000BD5'
NOM_FLOT_MQCACF	3030	X'00000BD6'
MQCACF_SUJET	3031	X'00000BD7'
MQCACF_PARENT_Q_MGR_NAME	3032	X'00000BD8'
ID_CORREL_MQCACF	3033	X'00000BD9'
MQCACF_HORODATAGE_PUBLICATION	3034	X'00000BDA'
DONNÉES_CHAÎNE_MQCACF	3035	X'00000BDB'
NOM_FLOT_SUPPORT_MQCACF	3036	X'00000BDC'
MQCACF_REG_TOPIC	3037	X'00000BDD'
HEURE DE REG_MQCACF	3038	X'00000BDE'
ID_UTILISATEUR_REG_MQCACF	3039	X'00000BDF'
MQCACF_CHILD_Q_MGR_NAME	3040	X'00000BE0'
MQCACF_REG_NOM_FLOT_TRAVAUX	3041	X'00000BE1'
MQCACF_REG_Q_MGR_NAME	3042	X'00000BE2'
MQCACF_REG_Q_NAME	3043	X'00000BE3'
ID_CORREL_REG_MQCACF	3044	X'00000BE4'
ID_UTILISATEUR_MQCACF_EVENT_USER_ID	3045	X'00000BE5'
MQCACF_NOM_OBJET	3046	X'00000BE6'
MQCACF_EVENT_Q_MGR	3047	X'00000BE7'
MQCACF_INFO_AUTH_NAMES	3048	X'00000BE8'
MQCACF_EVENT_APPL_IDENTITY	3049	X'00000BE9'
MQCACF_EVENT_APPL_NAME	3050	X'00000BEA'
MQCACF_EVENT_APPL_ORIGIN	3051	X'00000BEB'
MQCACF_NOM_ABONNEMENT	3052	X'00000BEC'

Tableau 40. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQCACF_REG_SOUS-NOM	3053	X'00000BED'
MQCACF_IDENTITÉ_ABONNEMENT	3054	X'00000BEE'
MQCACF_REG_SUB_IDENTITY	3055	X'00000BEF'
MQCACF_SUBSCRIPTION_USER_DATA	3056	X'00000BF0'
MQCACF_REG_SUB_USER_DATA	3057	X'00000BF1'
MQCACF_APPL_TAG	3058	X'00000BF2'
NOM_ENSEMBLE_DONNÉES_MQCACF	3059	X'00000BF3'
DATE DE DEMARRAGE MQCACF_UOW	3060	X'00000BF4'
MQCACF_UOW_HEURE DE DEMARRAGE	3061	X'00000BF5'
DATE DE DEMARRAGE MQCACF_UOW_LOG_START_DATE	3062	X'00000BF6'
MQCACF_UOW_LOG_START_TIME	3063	X'00000BF7'
MQCACF_UOW_LOG_EXTENT_NAME	3064	X'00000BF8'
MQCACF_PRINCIPAL_ENTITY_NAMES	3065	X'00000BF9'
MQCACF_GROUP_ENTITY_NAMES	3066	X'00000BFA'
NOM_PROFIL_AUTH_MQCACF	3067	X'00000BFB'
NOM_ENTITÉ_MQCACF	3068	X'00000BFC'
COMPOSANT_SERVICE_MQCACF_SERVICE	3069	X'00000BFD'
MQCACF_RESPONSE_Q_MGR_NAME	3070	X'00000BFE'
MQCACF_CURRENT_LOG_EXTENT_NAME	3071	X'00000BFF'
MQCACF_RESTART_LOG_EXTENT_NAME	3072	X'00000C00'
MQCACF_MEDIA_LOG_EXTENT_NAME	3073	X'00000C01'
CHEMIN_LOG_MQCACF	3074	X'00000C02'
MQCACF_COMMAND_MQSC	3075	X'00000C03'
MQCACF_Q_MGR_CPF	3076	X'00000C04'
MQCACF_USAGE_LOG_RBA	3078	X'00000C06'
MQCACF_USAGE_LOG_LRSN	3079	X'00000C07'
Portée de la commande MQCACF_COMMAND_SCOPE	3080	X'00000C08'
ID_AS_MQCACF	3081	X'00000C09'
MQCACF_PSB_NAME	3082	X'00000C0A'
ID_STP_MQCACF	3083	X'00000C0B'
NUMÉRO_TÂCHE_MQCACF	3084	X'00000C0C'
ID_TRANSACTION_MQCACF	3085	X'00000C0D'
MQCACF_Q_MGR_UOW_ID	3086	X'00000C0E'
MQCACF_NOM_ORIGINE	3088	X'00000C10'
MQCACF_ENV_INFO	3089	X'00000C11'
MQCACF_PROFIL_SÉCURITÉ_DE_SÉCURITÉ	3090	X'00000C12'
DATE_CONFIGURATION_MQCACF	3091	X'00000C13'
MQCACF_HEURE de configuration	3092	X'00000C14'
MQCACF_FROM_CF_NOM_STRUCTURE	3093	X'00000C15'

Tableau 40. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQCACF_TO_CF_STRUC_NAME	3094	X'00000C16'
MQCACF_CF_NOMS_STRUCTURES	3095	X'00000C17'
DATE D'ECHO_MQCACF_DATE	3096	X'00000C18'
DURE_ECHEC_MQCACF_HEURE	3097	X'00000C19'
DATE_SAUVEGARDE_MQCACF	3098	X'00000C1A'
MQCACF_HEURE de sauvegarde	3099	X'00000C1B'
NOM_SYSTÈME_MQCACF	3100	X'00000C1C'
MQCACF_CF_STRUC_BACKUP_START	3101	X'00000C1D'
MQCACF_CF_STRUC_BACKUP_END	3102	X'00000C1E'
MQCACF_CF_STRUC_LOG_Q_MGRS	3103	X'00000C1F'
CLASSE DE STOCKAGE_MQCACF_FROM_	3104	X'00000C20'
MQCACF_TO_STORAGE_CLASS	3105	X'00000C21'
MQCACF_STORAGE_NOMS_CLASSE_STOCKAGE	3106	X'00000C22'
NOM_DSG_MQCACF	3108	X'00000C24'
MQCACF_DB2_NAME	3109	X'00000C25'
MQCACF_SYSP_CMD_USER_ID	3110	X'00000C26'
GROUPE_MQCACF_SYSP_OTMA_GROUP	3111	X'00000C27'
MQCACF_SYSP_OTMA_MEMBRE	3112	X'00000C28'
MQCACF_SYSP_OTMA_DRU_EXIT	3113	X'00000C29'
MQCACF_SYSP_OTMA_TPIPE_PFX	3114	X'00000C2A'
MQCACF_SYSP_ARCHIVE_PFX1	3115	X'00000C2B'
MQCACF_SYSP_ARCHIVE_UNIT1	3116	X'00000C2C'
MQCACF_SYSP_LOG_CORREL_ID	3117	X'00000C2D'
MQCACF_SYSP_UNIT_VOLSER	3118	X'00000C2E'
MQCACF_SYSP_Q_MGR_TIME	3119	X'00000C2F'
MQCACF_SYSP_Q_MGR_DATE	3120	X'00000C30'
MQCACF_SYSP_Q_MGR_RBA	3121	X'00000C31'
MQCACF_SYSP_LOG_RBA	3122	X'00000C32'
SERVICE_MQCACF_SYSP_SERVICE	3123	X'00000C33'
NOM_LISTE_MQCACF_FROM_	3124	X'00000C34'
MQCACF_TO_LISTENER_NAME	3125	X'00000C35'
MQCACF_NOM_SERVICE_SOURCE	3126	X'00000C36'
MQCACF_TO_NOM_SERVICE	3127	X'00000C37'
MQCACF_DATE_DERNIÈRE_PUT_DATE	3128	X'00000C38'
MQCACF_DERNIER_TEMPS_PUT_TIME	3129	X'00000C39'
MQCACF_LAST_GET_DATE	3130	X'00000C3A'
MQCACF_LAST_GET_TIME	3131	X'00000C3B'
DATE_OPÉRATION_MQCACF	3132	X'00000C3C'
TEMPS-opération_mqcacf	3133	X'00000C3D'

Tableau 40. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQCACF_ACTIVITY_DESC	3134	X'00000C3E'
MQCACF_APPL_IDENTITY_DATA	3135	X'00000C3F'
MQCACF_APPL_ORIGIN_DATA	3136	X'00000C40'
DATE MQCACF_PUT_DATE	3137	X'00000C41'
MQCACF_HEURE	3138	X'00000C42'
MQCACF_RÉPONSE_À_Q	3139	X'00000C43'
MQCACF_REPLY_TO_Q_MGR	3140	X'00000C44'
MQCACF_RESOLVED_Q_NAME	3141	X'00000C45'
ID_STRUCF_MQCACF	3142	X'00000C46'
NOM_VALEUR_MQCACF	3143	X'00000C47'
DATE MQCACF_SERVICE_START_DATE	3144	X'00000C48'
MQCACF_SERVICE_START_TIME	3145	X'00000C49'
MQCACF_SYSP_OFFLINE_RBA	3146	X'00000C4A'
MQCACF_SYSP_ARCHIVE_PFX2	3147	X'00000C4B'
MQCACF_SYSP_ARCHIVE_UNIT2	3148	X'00000C4C'
MQCACF_TO_TOPIC_NAME	3149	X'00000C4D'
MQCACF_FROM_TOPIC_NAME	3150	X'00000C4E'
MQCACF_TOPIC_NAMES	3151	X'00000C4F'
MQCACF_NOM_SOUS-SYSTÈME	3152	X'00000C50'
MQCACF_DESTINATION_Q_MGR	3153	X'00000C51'
DESTINATION_MQCACF	3154	X'00000C52'
MQCACF_SUB_USER_ID	3156	X'00000C54'
DONNEES_SOUS-UTILISATEUR_MQCACF	3159	X'00000C57'
MQCACF_SUB_SELECTOR	3160	X'00000C58'
MQCACF_LAST_PUB_DATE	3161	X'00000C59'
MQCACF_DERNIER_HEURE_PUBLICATION	3162	X'00000C5A'
MQCACF_FROM_SUB_NAME	3163	X'00000C5B'
MQCACF_TO_SUB_NAME	3164	X'00000C5C'
MQCACF_LAST_MSG_TIME	3167	X'00000C5F'
MQCACF_LAST_MSG_DATE	3168	X'00000C60'
MQCACF_SUBSCRIPTION_POINT	3169	X'00000C61'
MQCACF_FILTRE	3170	X'00000C62'
MQCACF_NONE	3171	X'00000C63'
MQCACF_ADMIN_TOPIC_NAMES	3172	X'00000C64'
MQCACF_ROUTING_FINGER_PRINT	3173	X'00000C65'
MQCACF_APPL_DESC	3174	X'00000C66'
MQCACF_Q_MGR_START_DATE	3175	X'00000C67'
MQCACF_Q_MGR_START_TIME	3176	X'00000C68'
MQCACF_FROM_COMM_INFO_NAME	3177	X'00000C69'



Tableau 40. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQCACF_TO_COMM_INFO_NAME	3178	X'00000C6A'
MQCACF_CF_OFFLOAD_SIZE1	3179	X'00000C6B'
MQCACF_CF_OFFLOAD_SIZE2	3180	X'00000C6C'
MQCACF_CF_OFFLOAD_SIZE3	3181	X'00000C6D'
MQCACF_CF_SMDS_GENERIC_NAME	3182	X'00000C6E'
MQCACF_CF_SMDS	3183	X'00000C6F'
MQCACF_DATE_REPRISE	3184	X'00000C70'
MQCACF_TEMPS de reprise	3185	X'00000C71'
MQCACF_CF_SMDSCONN	3186	X'00000C72'
MQCACF_CF_NOM_STRUCTURE	3187	X'00000C73'
MQCACF_ALTERNATE_USERID	3188	X'00000C74'
MQCACF_CHAR_ATTRS	3189	X'00000C75'
NOM_Q_DYNAMIC_MQCACF	3190	X'00000C76'
MQCACF_NOM_HÔTE	3191	X'00000C77'
NOM_MQCACF_MQCB_	3192	X'00000C78'
MQCACF_OBJECT_CHAINE	3193	X'00000C79'
MQCACF_RESOLVED_LOCAL_Q_MGR	3194	X'00000C7A'
MQCACF_RESOLVED_LOCAL_Q_NAME	3195	X'00000C7B'
MQCACF_RESOLVED_OBJECT_STRING	3196	X'00000C7C'
MQCACF_RESOLVED_Q_MGR	3197	X'00000C7D'
CHAÎNE_SÉLECTION_MQCACF	3198	X'00000C7E'
MQCACF_XA_INFO	3199	X'00000C7F'
MQCACF_FONCTION_APPL_MQCACF	3200	X'00000C80'
MQCACF_XQH_REMOTE_Q_NAME	3201	X'00000C81'
MQCACF_XQH_REMOTE_Q_MGR	3202	X'00000C82'
MQCACF_XQH_HEURE	3203	X'00000C83'
DATE MQCACF_XQH_PUT_DATE	3204	X'00000C84'
MQCACF_EXCL_MESSAGES_OPÉRATEUR_NON	3205	X'00000C85'
MQCACF_CSP_IDENTIFICATEUR_UTILISATEUR	3206	X'00000C86'
MQCACF_AMQP_CLIENT_ID	3207	X'00000C87'
MQCACF_ARCHIVE_LOG_NOM_EXTENSION	3208	X'00000C88'
MQCACF_APPL_IMMOVABLE_DATE	3209	X'00000C89'
MQCACF_APPL_IMMOVABLE_TIME	3210	X'00000C8A'
MQCACF_NHA_NOM_INSTANCE	3211	X'00000C8B'
MQCACF_LAST_USED	3211	X'00000C8B'

## MQCACH\_\* (types de paramètre de canal de caractères de format de commande)

Tableau 41. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCACH_FIRST	3501	X'00000DAD'
MQCACH_CHANNEL_NAME	3501	X'00000DAD'
DES_MQCACH_DESC	3502	X'00000DAE'
MQCACH_MODE_NAME	3503	X'00000DAF'
MQCACH_NOM_TP	3504	X'00000DB0'
MQCACH_NOM_Q_XMIT_CACHE	3505	X'00000DB1'
MQCACH_NOM_CONNEXION	3506	X'00000DB2'
MQCACH_MCA_NAME	3507	X'00000DB3'
MQCACH_SEC_NOM_SORTIE	3508	X'00000DB4'
MQCACH_MSG_EXIT_NAME	3509	X'00000DB5'
MQCACH_SEND_NOM_SORTIE	3510	X'00000DB6'
MQCACH_RCV_NOM_SORTIE	3511	X'00000DB7'
MQCACH_CHANNEL_NAMES	3512	X'00000DB8'
DONNEES_UTILISATEUR_EXIT_MQCACH_SEC_	3513	X'00000DB9'
DONNEES_USER_EXIT_MQCACH_MSG_	3514	X'00000DBA'
DONNEES_UTILISATEUR_EXIT_MQCACH_SEND_	3515	X'00000DBB'
DONNEES_UTILISATEUR_EXIT_MQCACH_RCV_	3516	X'00000DBC'
ID_UTILISATEUR_MQCACHE	3517	X'00000DBD'
MOT_DE_PASSE_MQCACH_CACHE	3518	X'00000DBE'
ADRESSE_LOCAL_MQCACH_LOCAL_ADDRESS	3520	X'00000DC0'
MQCACH_NOM_LOCAL_LOCAL	3521	X'00000DC1'
MQCACH_LAST_MSG_TIME	3524	X'00000DC4'
MQCACH_LAST_MSG_DATE	3525	X'00000DC5'
ID_UTILISATEUR_MCA_MQCACH_	3527	X'00000DC7'
MQCACH_CHANNEL_START_TIME	3528	X'00000DC8'
MQCACH_CHANNEL_START_DATE	3529	X'00000DC9'
MQCACH_MCA_NOM_TRAVAIL	3530	X'00000DCA'
MQCACH_LAST_LUWID	3531	X'00000DCB'
MQCACH_CURRENT_LUWID	3532	X'00000DCC'
MQCACH_FORMAT_NAME	3533	X'00000DCD'
NOM_EXIT_MR_MQCACH_	3534	X'00000DCE'
DONNEES_UTILISATEUR_EXIT_MQCACH_MR_DONNEES_UTILISAT EUR	3535	X'00000DCF'
MQCACH_SSL_CIPHER_SPEC	3544	X'00000DD8'
NOM_PE_SSL_MQCACH_	3545	X'00000DD9'
MQCACH_SSL_HANDSHAKE_STAGE	3546	X'00000DDA'
MQCACH_SSL_SHORT_PEER_NAME	3547	X'00000ddb'
MQCACH_REMOTE_APPL_TAG	3548	X'00000DDC'

Tableau 41. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQCACH_SSL_CERT_USER_ID	3549	X'00000DDD'
MQCACH_SSL_CERT_NOM DE DIFFUSION	3550	X'00000DDE'
MQCACH_LU_NAME	3551	X'00000DDF'
ADRESSE_IP_MQCACH_IP	3552	X'00000DE0'
NOM_TCP_MQCACHE	3553	X'00000DE1'
NOM_LISTE_MQCACHE	3554	X'00000DE2'
MQCACH_LISTENER_DESC	3555	X'00000DE3'
DATE MQCACH_LISTENER_START_DATE	3556	X'00000DE4'
HEURE-DEMARRAGE-LISTE_MQCACHE	3557	X'00000DE5'
MQCACH_SSL_KEY_RESET_DATE	3558	X'00000DE6'
MQCACH_SSL_KEY_RESET_TIME	3559	X'00000DE7'
MQCACH_LAST_USED	3559	X'00000DE7'

### **MQCADSD\_\* (Descripteurs ADS d'en-tête d'informations CICS)**

Tableau 42. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCADSD_NONE	0	X'00000000'
MQCADSD_SEND	1	X'00000001'
MQCADSD_RECV	16	X'00000010'
MQCADSD_MSGFORMAT	256	X'00000100'

### **MQCAFTY\_\* (Valeurs d'affinité de connexion)**

Tableau 43. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCAFTY_NONE	0	X'00000000'
MQCAFTY_PREFERENTE_PRÉFÉRÉE	1	X'00000001'

### **MQCAMO\_\* (format de commande-Types de paramètre de surveillance des caractères)**

Tableau 44. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCAMO_FIRST	2701	X'00000A8D'
DATE MQCAMO_CLOSE_DATE	2701	X'00000A8D'
HEURE MQCAMO_CLOSE	2702	X'00000A8E'
DATE MQCAMO_CONN_DATE	2703	X'00000A8F'
HEURE_CONNEXION MQCAMO_MQ	2704	X'00000A90'
DATE MQCAMO_DISC_DATE	2705	X'00000A91'
HEURE-DISC_MQCAMO_	2706	X'00000A92'
DATE MQCAMO_FIN	2707	X'00000A93'
Heure_END_MQCAMO	2708	X'00000A94'

Tableau 44. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
DATE MQCAMO_OPEN_DATE	2709	X'00000A95'
HEURE DE L'OPEN_MQCAMO	2710	X'00000A96'
DATE MQCAMO_START_DATE	2711	X'00000A97'
HEURE DE DEMARRAGE MQCAMO_DEMARRAGE	2712	X'00000A98'
MQCAMO_LAST_USED	2712	X'00000A98'

### MQCBC\_\* (structure des constantes MQCBC)

Tableau 45. Structures de constantes	
Nom	Structure
MQCBC_STRUC_ID	"CBC~"
MQCBC_STRUC_ID_ARRAY	'C','B','C','~'

**Remarque :** Le symbole ~ représente un caractère blanc unique.

Tableau 46. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCBC_VERSION_1	1	X'00000001'
MQCBC_CURRENT_VERSION	1	X'00000001'

### MQCBCF\_\* (indicateurs de constantes MQCBC)

Tableau 47. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCBCF_AUCUN	0	X'00000000'
MQCBCF_READA_BUFFER_EMPTY	1	X'00000001'

### MQCBCT\_\* (type de rappel de constantes MQCBC)

Tableau 48. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
Appel MQCBCT_START_CALL	1	X'00000001'
MQCBCT_STOP_CALL	2	X'00000002'
APPEL MQCBCT_REGISTER_CALL	3	X'00000003'
MQCBCT_DEREGISTER_CALL	4	X'00000004'
MQCBCT_EVENT_CALL	5	X'00000005'
MQCBCT_MSG_REMOVED	6	X'00000006'
MQCBCT_MSG_NOT_REMOVED	7	X'00000007'

### MQCBD\_\* (structure des constantes MQCBD)

Tableau 49. Structures de constantes	
Nom	Structure
ID MQCBD_STRUC_ID	"CBD~"
MQCBD_STRUC_ID_ARRAY	'C','B','D','~'

**Remarque :** Le symbole – représente un caractère blanc unique.

Tableau 50. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCBD_VERSION_1	1	X'00000001'
MQCBD_CURRENT_VERSION	1	X'00000001'

### MQCBDO\_\* (Options de rappel des constantes MQCBD)

Tableau 51. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCBDO_AUCUN	0	X'00000000'
Appel MQCBDO_START_CALL	1	X'00000001'
Appel MQCBDO_STOP_CALL	4	X'00000004'
APPEL MQCBDO_REGISTER_CALL	256	X'00000100'
MQCBDO_DEREGISTER_CALL	512	X'00000200'
MQCBDO_ECHEC_IF_QUIESCING	8192	X'00002000'

### MQCBO\_\* (options de création de sac pour mqCreateBag)

Tableau 52. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCBO_AUCUN	0	X'00000000'
MQCBO_USER_BAG	0	X'00000000'
MQCBO_ADMIN_BAG	1	X'00000001'
SAC de commande mqcbo_commande	16	X'00000010'
MQCBO_SYSTEM_BAG	32	X'00000020'
MQCBO_GROUP_BAG	64	X'00000040'
MQCBO_LIST_FORM_ALLOWED	2	X'00000002'
MQCBO_LIST_FORM_INHIBÉ	0	X'00000000'
MQCBO_REORDER_AS_REQUIRED	4	X'00000004'
MQCBO_DO_NOT_REORDER	0	X'00000000'
MQCBO_CHECK_SELTEURS	8	X'00000008'
MQCBO_DO_NOT_CHECK_SELECTEURS	0	X'00000000'

### MQCBT\_\* (constantes MQCBD Il s'agit du type de la fonction de rappel)

Tableau 53. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCBT_MESSAGE_CONSUMER	1	X'00000001'
MQCBT_EVENT_HANDLER	2	X'00000002'

## MQCC\_\* (codes achèvement)

Tableau 54. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCC_OK	0	X'00000000'
MQCC_WARNING	1	X'00000001'
MQCC_FAILED	2	X'00000002'
MQCC_INCONNU	-1	X'FFFFFFFF'

## MQCCSI\_\* (Identificateurs de jeu de caractères codés)

Tableau 55. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCCSI_UNDEFINI	0	X'00000000'
MQCCSI_DEFAULT	0	X'00000000'
MQCCSI_Q_DIR	0	X'00000000'
MQCCSI_HÉRITER	-2	X'FFFFFFFE'
MQCCSI_IMBRIQUÉ	-1	X'FFFFFFFF'
MQCCSI_APPL	-3	X'FFFFFFFD'

## MQCCT\_\* (options de tâche conversationnelle d'en-tête d'information CICS)

Tableau 56. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCCT_OUI	1	X'00000001'
MQCCT_NO	0	X'00000000'

## MQCD\_\* (structure de définition de canal)

Tableau 57. Valeurs des constantes






Nom	Valeur décimale	Valeur hexadécimale
MQCD_VERSION_1	1	X'00000001'
MQCD_VERSION_2	2	X'00000002'
MQCD_VERSION_3	3	X'00000003'
MQCD_VERSION_4	4	X'00000004'
MQCD_VERSION_5	5	X'00000005'
MQCD_VERSION_6	6	X'00000006'
MQCD_VERSION_7	7	X'00000007'
MQCD_VERSION_8	8	X'00000008'
MQCD_VERSION_9	9	X'00000009'
MQCD_VERSION_10	10	X'0000000A'
 MQCD_VERSION_11	11	X'0000000B'
 MQCD_CURRENT_VERSION	11	X'0000000B'
 MQCD_VERSION_12	12	X'0000000C'

Tableau 57. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
 MQCD_CURRENT_VERSION	12	X'0000000C'
MQCD_LENGTH_4	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_5	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_6	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_7	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_8	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_9	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_10	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_11	(value differs by platform or version)	(value differs by platform or version)
 MQCD_LENGTH_12	(value differs by platform or version)	(value differs by platform or version)
MQCD_LONGUEUR_EN_COURS	(value differs by platform or version)	(value differs by platform or version)

### MQCDC\_\* (Conversion de données de canal)

Tableau 58. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCDC_SENDER_CONVERSION	1	X'00000001'
MQCDC_NO_SENDER_CONVERSION	0	X'00000000'

### MQCERT\_\* (Type de règle de validation de certificat)

MQ_CERT_VAL_POLICY_DEFAULT	0	X'00000000'
MQ_CERT_VAL_POLICY_ANY	0	X'00000000'
MQ_CERT_VAL_POLICY_RFC5280	1	X'00000001'

### MQCF\_\* (indicateurs de capacité)

Tableau 59. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCF_AUCUN	0	X'00000000'
LIST_LISTES MQCF_DIST	1	X'00000001'

### MQCFAC\_\* (fonction d'en-tête d'informations CICS)

Tableau 60. Noms et valeurs de constante

Nom	Valeur hexadécimale
MQCFAC_NONE	X'00...00' (8 valeurs nulles)

Tableau 60. Noms et valeurs de constante (suite)	
Nom	Valeur hexadécimale
MQCFAC_NON_ARRAY	'\0', '\0', ... (8 valeurs nulles)

### **MQCFBF\_\* (structure de paramètre de filtre de chaîne d'octet de format de commande)**

Tableau 61. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCFBF_STRUC_LENGTH_FIXED	20	X'00000014'

### **MQCFBS\_\* (structure des paramètres de chaîne d'octets de format de commande)**

Tableau 62. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCFBS_STRUC_LENGTH_FIXED	16	X'00000010'

### **MQCFC\_\* (options de contrôle d'en-tête de format de commande)**

Tableau 63. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCFC_LAST	1	X'00000001'
MQCFC_NOT_LAST	0	X'00000000'

### **MQCFGR\_\* (Structure des paramètres de groupe de format de commande)**

Tableau 64. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCFGR_LONGUEUR_STRUCTURE	16	X'00000010'

### **MQCFH\_\* (structure d'en-tête de format de commande)**

Tableau 65. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
LONGUEUR_STRUC_MQCFH	36	X'00000024'
MQCFH_VERSION_1	1	X'00000001'
MQCFH_VERSION_2	2	X'00000002'
MQCFH_VERSION_3	3	X'00000003'
MQCFH_CURRENT_VERSION	3	X'00000003'

### **MQCFIF\_\* (structure de paramètre de filtre d'entier de format de commande)**

Tableau 66. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
LONGUEUR_STRUC_MQCFIF_LENGTH	20	X'00000014'



## **MQCFIL\_\* (Structure des paramètres de liste d'entiers de format de commande)**

<i>Tableau 67. Valeurs des constantes</i>		
<b>Nom</b>	<b>Valeur décimale</b>	<b>Valeur hexadécimale</b>
MQCFIL_STRUC_LENGTH_FIXED	16	X'00000010'

## **MQCFIL64\_\* (structure de paramètres de liste d'entiers 64 bits de format de commande)**

<i>Tableau 68. Valeurs des constantes</i>		
<b>Nom</b>	<b>Valeur décimale</b>	<b>Valeur hexadécimale</b>
MQCFIL64_STRUC_LENGTH_FIXED	16	X'00000010'

## **MQCFIN\_\* (Structure des paramètres de type entier de format de commande)**

<i>Tableau 69. Valeurs des constantes</i>		
<b>Nom</b>	<b>Valeur décimale</b>	<b>Valeur hexadécimale</b>
MQCFIN_LONGUEUR_STRUCTURE	16	X'00000010'

## **MQCFIN64\_\* (structure de paramètres d'entier 64 bits de format de commande)**

<i>Tableau 70. Valeurs des constantes</i>		
<b>Nom</b>	<b>Valeur décimale</b>	<b>Valeur hexadécimale</b>
MQCFIN64_STRUC_LENGTH	24	X'00000018'

## **MQCFO\_\* (Options de régénération du format de commande et options de suppression de files d'attente)**

### **Options du référentiel d'actualisation de format de commande**

<i>Tableau 71. Valeurs des constantes</i>		
<b>Nom</b>	<b>Valeur décimale</b>	<b>Valeur hexadécimale</b>
MQCFO_REFRESH_REPOSITORY_YES	1	X'00000001'
MQCFO_REFRESH_REPOSITORY_NO	0	X'00000000'

### **Format de commande-Options de suppression de files d'attente**

<i>Tableau 72. Valeurs des constantes</i>		
<b>Nom</b>	<b>Valeur décimale</b>	<b>Valeur hexadécimale</b>
MQCFO_REMOVE_QUEUES_YES	1	X'00000001'
MQCFO_REMOVE_QUEUES_NO	0	X'00000000'

## MQCFOP\_\* (Opérateurs de filtre de format de commande)

Tableau 73. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCFOP_LESS	1	X'00000001'
MQCFOP_EQUAL	2	X'00000002'
MQCFOP_SUPÉRIEUR	4	X'00000004'
MQCFOP_NOT_LESS	6	X'00000006'
MQCFOP_NOT_EQUAL	5	X'00000005'
MQCFOP_NON_SUPÉRIEUR	3	X'00000003'
MQCFOP_LIKE	18	X'00000012'
MQCFOP_NOT_LIKE	21	X'00000015'
MQCFOP_CONTIENT	10	X'0000000A'
MQCFOP_EXCLUDES	13	X'0000000D'
MQCFOP_CONTAINS_GEN	26	X'0000001A'
MQCFOP_EXCLUDES_GEN	29	X'0000001D'

## MQCFR\_\* (capacité de reprise de l'unité de couplage)

Tableau 74. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCFR_OUI	1	X'00000001'
MQCFR_NO	0	X'00000000'

## MQCFSF\_\* (Structure de paramètre de filtre de chaîne de format de commande)

Tableau 75. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCFSF_STRUC_LENGTH_FIXED	24	X'00000018'

## MQCFSL\_\* (Structure de paramètre de liste de chaînes de format de commande)

Tableau 76. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCFSL_STRUC_LENGTH_FIXED	24	X'00000018'

## MQCFST\_\* (Structure de paramètre de chaîne de format de commande)

Tableau 77. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCFST_STRUC_LENGTH_FIXED	20	X'00000014'

## MQCFSTATUS\_\* (Statut de l'unité de couplage de format de commande)

Tableau 78. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCFSTATUS_NOT_FOUND	0	X'00000000'
MQCFSTATUS_ACTIVE	1	X'00000001'
MQCFSTATUS_DANS_RECUPERER	2	X'00000002'
MQCFSTATUS_DANS_LA sauvegarde	3	X'00000003'
MQCFSTATUS_ECHEC	4	X'00000004'
MQCFSTATUS_NONE	5	X'00000005'
MQCFSTATUS_INCONNU	6	X'00000006'
MQCFSTATUS_ADMIN_INCOMPLET	20	X'00000014'
MQCFSTATUS_NEVER_USED	21	X'00000015'
MQCFSTATUS_NO_BACKUP	22	X'00000016'
MQCFSTATUS_NOT_ECHEC	23	X'00000017'
MQCFSTATUS_NON_RÉCUPÉRABLE	24	X'00000018'
MQCFSTATUS_XES_ERREUR	25	X'00000019'

## MQCFT\_\* (types de format de commande de la structure)

Tableau 79. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCFT_AUCUN	0	X'00000000'
MQCFT_COMMAND (COMMANDE MQ)	1	X'00000001'
MQCFT_REPONSE	2	X'00000002'
MQCFT_ENTIER	3	X'00000003'
MQCFT_CHAINE	4	X'00000004'
MQCFT_INTEGER_LIST	5	X'00000005'
MQCFT_STRING_LIST	6	X'00000006'
MQCFT_EVENT	7	X'00000007'
Utilisateur_MQCF	8	X'00000008'
MQCFT_BYTE_CHAINE	9	X'00000009'
MQCFT_TRACE_ROUTE	10	X'0000000A'
RAPPORT MQCF	12	X'0000000C'
MQCFT_INTEGER_FILTER	13	X'0000000D'
FILTRE_CHAÎNE_MQCFT_FILTER	14	X'0000000E'
MQCFT_BYTE_STRING_FILTER	15	X'0000000F'
MQCFT_COMMAND_XR	16	X'00000010'
MQCFT_XR_MSG	17	X'00000011'
MQCFT_XR_ITEM	18	X'00000012'
MQCFT_XR_SUMMARY	19	X'00000013'
GROUPE MQCF	20	X'00000014'
STATISTIQUES MQCF	21	X'00000015'

Tableau 79. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQCFT_ACCOUNTING	22	X'00000016'
MQCFT_INTEGER64	23	X'00000017'
MQCFT_INTEGER64_LIST	25	X'00000019'

### MQCFTYPE\_\* (types d'unité de couplage de format de commande)

Tableau 80. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCFTYPE_APPL	0	X'00000000'
MQCFTYPE_ADMIN	1	X'00000001'

### MQCFUNC\_\* (fonctions d'en-tête d'informations CICS )

Tableau 81. Structures de constantes	
Nom	Structure
MQCFUNC_MQCONN	"CONN"
MQCFUNC_MQGET	"GET~"
MQCFUNC_MQINQ	"INQ~"
MQCFUNC_MQOPEN	"OPEN"
MQCFUNC_MQPUT	"PUT~"
MQCFUNC_MQPUT1	"PUT1"
MQCFUNC_NONE	"~~~~"
MQCFUNC_MQCONN_ARRAY	'C','O','N','N'
MQCFUNC_MQGET_ARRAY	'G','E','T','~'
MQCFUNC_MQINQ_ARRAY	'I','N','Q','~'
MQCFUNC_MQOPEN_ARRAY	'O','P','E','N'
MQCFUNC_MQPUT_ARRAY	'P','U','T','~'
MQCFUNC_MQPUT1_ARRAY	'P','U','T','1'
MQCFUNC_NON_ARRAY	'~','~','~','~'

**Remarque :** Le symbole ~ représente un caractère blanc unique.

### MQCGWI\_\* (en-tête d'information CICS Get Wait Interval)

Tableau 82. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCGWI_VALEUR PAR DEFALT	-2	X'FFFFFFFE'

### MQCHAD\_\* (définition automatique de canal)

Tableau 83. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCHAD_DISABLED	0	X'00000000'
MQCHAD_ENABLED	1	X'00000001'

## MQCHIDS\_\* (Format de commande-Etat en attente de validation)

Tableau 84. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCHIDS_NOT_INDOUBT	0	X'00000000'
MQCHIDS_INDOUBT	1	X'00000001'

## MQCHLD\_\* (Dispositions de canal de format de commande)

Tableau 85. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCHLD_ALL	-1	X'FFFFFFFF'
MQCHLD_VALEUR PAR DEFAUT	1	X'00000001'
MQCHLD_SHARED	2	X'00000002'
MQCHLD_PRIVÉ	4	X'00000004'
MQCHLD_FIXSHARED	5	X'00000005'

## MQCHS\_\* (Statut du canal du format de commande)

Tableau 86. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCHS_INACTIVE	0	X'00000000'
MQCHS_BINDING	1	X'00000001'
MQCHS_STARTING	2	X'00000002'
MQCHS_RUNNING	3	X'00000003'
MQCHS_STOPPING	4	X'00000004'
MQCHS_RETRYING	5	X'00000005'
MQCHS_STOPPED	6	X'00000006'
MQCHS_REQUESTING	7	X'00000007'
MQCHS_PAUSED	8	X'00000008'
MQCHS_INITIALIZING	13	X'0000000D'
MQCHS_SWITCHING	14	X'0000000E'

## MQCHSH\_\* (options de redémarrage partagé du canal de format de commande)

Tableau 87. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCHSH_RESTART_NO	0	X'00000000'
MQCHSH_RESTART_YES	1	X'00000001'

## MQCHSR\_\* (Options d'arrêt du canal de format de commande)

Tableau 88. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCHSR_STOP_NOT_REQUESTED	0	X'00000000'

Tableau 88. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQCHSR_STOP_DEMANDÉ	1	X'00000001'

### MQCHSSTATE\_\* (Sous-états du canal de format de commande)

Tableau 89. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCHSSTATE_AUTRES	0	X'00000000'
MQCHSSTATE_END_OF_BATCH	100	X'00000064'
MQCHSSTATE_ENVOI	200	X'000000C8'
MQCHSSTATE_RECEIVING	300	X'0000012C'
MQCHSSTATE_SÉRIALISATION	400	X'00000190'
MQCHSSTATE_RESYNCHING	500	X'000001F4'
MQCHSSTATE_PULSATION	600	X'00000258'
MQCHSSTATE_IN_SCYEXIT	700	X'000002BC'
MQCHSSTATE_IN_RCVEXIT	800	X'00000320'
MQCHSSTATE_IN_SENDEXIT	900	X'00000384'
MQCHSSTATE_IN_MSGEXIT	1000	X'000003E8'
MQCHSSTATE_DANS_MREXIT	1100	X'0000044C'
MQCHSSTATE_IN_CHADEXIT	1200	X'000004B0'
MQCHSSTATE_NET_CONNEXION	1250	X'000004E2'
MQCHSSTATE_SSL_HANDSHAKE	1300	X'00000514'
MQCHSSTATE_NAME_SERVER	1400	X'00000578'
MQCHSSTATE_IN_MQPUT	1500	X'000005DC'
MQCHSSTATE_IN_MQGET	1600	X'00000640'
MQCHSSTATE_IN_MQI_CALL	1700	X'000006A4'
MQCHSSTATE_COMPRESSION	1800	X'00000708'

### MQCHT\_\* (types de canal)

Tableau 90. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
EXPÉDITEUR_MQCH	1	X'00000001'
SERVEUR_MQ	2	X'00000002'
MQCHT_RECEIVER	3	X'00000003'
DEMANDE_MQCHT_DEMANDEUR	4	X'00000004'
MQCHT_ALL	5	X'00000005'
MQCHT_CLNTCONN	6	X'00000006'
MQCHT_SVRCONN	7	X'00000007'
MQCHT_CLUSRCVR	8	X'00000008'
MQCHT_CLUSSDR	9	X'00000009'

## MQCHTAB\_\* (Types de table de canal de format de commande)

Tableau 91. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCHTAB_Q_MGR	1	X'00000001'
MQCHTAB_CLNTCONN	2	X'00000002'

## MQCI\_\* (identificateur de corrélation)

Tableau 92. Noms et valeurs de constante

Nom	Valeur
MQCI_NONE	X'00...00' (24 valeurs nulles)
MQCI_NON_ARRAY	'\0', '\0', ... (24 valeurs nulles)
MQCI_NOUVEAU_SESSION	X'414D5121...'
MQCI_NOUVELLE_MATRICE de sessions	'\x41', '\x4D', '\51', '\x21', ...

## MQCIH\_\* (indicateurs et structure d'en-tête d'information CICS)

### Structure d'en-tête d'informations CICS

Tableau 93. Structures de constantes

Nom	Structure
ID_STRUC_MQCIH	"CIH~"
MQCIH_STRUC_ID_ARRAY	'C', 'I', 'H', '~'

**Remarque :** Le symbole ~ représente un caractère blanc unique.

Tableau 94. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCIH_VERSION_1	1	X'00000001'
MQCIH_VERSION_2	2	X'00000002'
VERSION_MQCIH_CURRENT_VERSION	2	X'00000002'
MQCIH_LENGTH_1	164	X'000000A4'
MQCIH_LENGTH_2	180	X'000000B4'
MQCIH_LONGUEUR_EN_COURS	180	X'000000B4'

### Indicateurs d'en-tête d'informations CICS

Tableau 95. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCIH_NONE	0	X'00000000'
EXPIRATION_MOT_DE_PASSE_MQCIH	1	X'00000001'
MQCIH_EXPIRATION non limitée	0	X'00000000'
MQCIH_REPLY_WITHOUT_NULLS	2	X'00000002'
MQCIH_RÉPONSE_AVEC_NULS	0	X'00000000'
MQCIH_SYNC_ON_RETURN	4	X'00000004'
MQCIH_NO_SYNC_ON_RETURN	0	X'00000000'

## MQCLCT\_\* (types de cache de cluster)

Tableau 96. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCLCT_STATIC	0	X'00000000'
MQCLCT_DYNAMIQUE	1	X'00000001'

## MQCLRS\_\* (format de commande-Effacement de la portée de la chaîne de rubrique)

Tableau 97. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCLRS_LOCAL	1	X'00000001'
MQCLRS_GLOBAL	2	X'00000002'

## MQCLRT\_\* (Format de commande-Type de chaîne de rubrique d'effacement)

Tableau 98. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCLRT_RETENU	1	X'00000001'

## MQCLT\_\* (types de lien d'en-tête d'information CICS)

Tableau 99. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
PROGRAMME MQCL	1	X'00000001'
TRANSACTION MQCL	2	X'00000002'

## MQCLWL\_\* (charge de travail du cluster)

Tableau 100. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCLWL_USEQ_LOCAL	0	X'00000000'
MQCLWL_USEQ_ANY	1	X'00000001'
MQCLWL_USEQ_AS_Q_MGR	-3	X'FFFFFFFD'

## MQCLXQ\_\* (Type de file d'attente de transmission de cluster)

MQCLXQ\_\* sont les valeurs que vous pouvez définir dans l'attribut de gestionnaire de files d'attente DEFCLXQ. L'attribut **DEFCLXQ** contrôle la file d'attente de transmission sélectionnée par défaut par les canaux émetteurs de cluster pour extraire les messages et les envoyer aux canaux récepteurs de cluster.

Tableau 101. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCLXQ_SCTQ	0	X'00000000'
MQCLXQ_CHANNEL	1	X'00000001'

### Référence associée

«DefClusterXmitQueueType (MQLONG)», à la page 855



L'attribut DefClusterXmitQueue contrôle la file d'attente de transmission sélectionnée par défaut par les canaux émetteurs de cluster à partir de laquelle les messages doivent être envoyés aux canaux récepteurs de cluster.

[Modifier un gestionnaire de files d'attente](#)

[Consulter les gestionnaires de files d'attente](#)

[Interroger le gestionnaire de files d'attente \(réponse\)](#)

«MQINQ-Attributs d'objet Inquire», à la page 737

L'appel MQINQ renvoie un tableau d'entiers et un ensemble de chaînes de caractères contenant les attributs d'un objet.

## MQCMD\_ \* (codes de commande)

Tableau 102. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCMD_NONE	0	X'00000000'
MQCMD_CHANGE_Q_MGR	1	X'00000001'
MQCMD_INQUIRE_Q_MGR	2	X'00000002'
MQCMD_CHANGE_PROCESSUS	3	X'00000003'
MQCMD_COPY_PROCESS	4	X'00000004'
MQCMD_CREATE_PROCESS	5	X'00000005'
MQCMD_DELETE_PROCESS	6	X'00000006'
MQCMD_INQUIRE_PROCESS	7	X'00000007'
MQCMD_CHANGE_Q	8	X'00000008'
MQCMD_CLEAR_Q	9	X'00000009'
MQCMD_COPY_Q	10	X'0000000A'
MQCMD_CREATE_Q	11	X'0000000B'
MQCMD_DELETE_Q	12	X'0000000C'
MQCMD_INQUIRE_Q	13	X'0000000D'
MQCMD_REFRESH_Q_MGR	16	X'00000010'
MQCMD_RESET_Q_STATS	17	X'00000011'
MQCMD_INQUIRE_Q_NAMES	18	X'00000012'
MQCMD_INQUIRE_PROCESS_NAMES	19	X'00000013'
MQCMD_INQUIRE_CHANNEL_NAMES	20	X'00000014'
MQCMD_CHANGE_CHANNEL	21	X'00000015'
CANAL_COPIE_MQCMD_	22	X'00000016'
MQCMD_CREATE_CHANNEL	23	X'00000017'
MQCMD_DELETE_CHANNEL	24	X'00000018'
MQCMD_INQUIRE_CHANNEL	25	X'00000019'
MQCMD_PING_CHANNEL	26	X'0000001A'
CANAL_REINITIALISATION_MQCMD_CHANNEL	27	X'0000001B'
MQCMD_START_CHANNEL	28	X'0000001C'
MQCMD_STOP_CHANNEL	29	X'0000001D'
MQCMD_START_CHANNEL_INIT	30	X'0000001E'
MQCMD_START_CHANNEL_LISTENER	31	X'0000001F'

Tableau 102. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQCMD_CHANGE_NAMELIST	32	X'00000020'
MQCMD_COPY_NAMELIST	33	X'00000021'
MQCMD_CREATE_NAMELIST	34	X'00000022'
MQCMD_DELETE_NAMELIST	35	X'00000023'
MQCMD_INQUIRE_NAMELIST	36	X'00000024'
MQCMD_INQUIRE_NAMELIST_NAMES	37	X'00000025'
MQCMD_ECHAPPEMENT	38	X'00000026'
MQCMD_RESOLVE_CHANNEL	39	X'00000027'
MQCMD_PING_Q_MGR	40	X'00000028'
MQCMD_INQUIRE_Q_STATUS	41	X'00000029'
MQCMD_INQUIRE_CHANNEL_STATUS	42	X'0000002A'
MQCMD_CONFIG_EVENT	43	X'0000002B'
MQCMD_Q_MGR_EVENT	44	X'0000002C'
MQCMD_PERFM_EVENT	45	X'0000002D'
MQCMD_CHANNEL_EVENT	46	X'0000002E'
MQCMD_DELETE_PUBLICATION	60	X'0000003C'
MQCMD_DÉSENREGISTRE_DIFFUSEUR de publications	61	X'0000003D'
MQCMD_DÉSENREGISTRE_ABONNÉ	62	X'0000003E'
MQCMD_PUBLICATION	63	X'0000003F'
MQCMD_REGISTER_PUBLISHER	64	X'00000040'
ABONNÉ_REGISTER_MQCMD_	65	X'00000041'
MQCMD_REQUEST_UPDATE	66	X'00000042'
MQCMD_BROKER_INTERNAL	67	X'00000043'
MQCMD_ACTIVITY_MSG	69	X'00000045'
MQCMD_INQUIRE_CLUSTER_Q_MGR	70	X'00000046'
MQCMD_RESUME_Q_MGR_CLUSTER	71	X'00000047'
MQCMD_SUSPEND_Q_MGR_CLUSTER	72	X'00000048'
MQCMD_REFRESH_CLUSTER	73	X'00000049'
MQCMD_RESET_CLUSTER	74	X'0000004A'
MQCMD_TRACE_ROUTE	75	X'0000004B'
SECURITE MQCMD_REFRESH_SECURITE	78	X'0000004E'
MQCMD_CHANGE_AUTH_INFO	79	X'0000004F'
MQCMD_COPY_AUTH_INFO	80	X'00000050'
MQCMD_CREATE_AUTH_INFO	81	X'00000051'
MQCMD_DELETE_AUTH_INFO	82	X'00000052'
MQCMD_INQUIRE_AUTH_INFO	83	X'00000053'
MQCMD_INQUIRE_AUTH_INFO_NAMES	84	X'00000054'
MQCMD_INQUIRE_CONNECTION	85	X'00000055'
MQCMD_STOP_CONNECTION	86	X'00000056'

Tableau 102. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQCMD_INQUIRE_AUTH_RECS	87	X'00000057'
MQCMD_INQUIRE_ENTITY_AUTH	88	X'00000058'
MQCMD_DELETE_AUTH_REC	89	X'00000059'
MQCMD_SET_AUTH_REC	90	X'0000005A'
MQCMD_LOGGER_EVENT	91	X'0000005B'
MQCMD_RESET_Q_MGR	92	X'0000005C'
MQCMD_CHANGE_LISTENER	93	X'0000005D'
MQCMD_COPY_LISTENER	94	X'0000005E'
MQCMD_CREATE_LISTENER	95	X'0000005F'
MQCMD_DELETE_LISTENER	96	X'00000060'
MQCMD_INQUIRE_LISTENER	97	X'00000061'
MQCMD_INQUIRE_LISTENER_STATUS	98	X'00000062'
ÉVÉNEMENT_COMMANDE_MQCMD_	99	X'00000063'
MQCMD_CHANGE_SECURITY	100	X'00000064'
MQCMD_CHANGE_CF_STRUC	101	X'00000065'
MQCMD_CHANGE_STG_CLASS	102	X'00000066'
MQCMD_CHANGE_TRACE	103	X'00000067'
MQCMD_ARCHIVE_LOG	104	X'00000068'
MQCMD_BACKUP_CF_STRUC	105	X'00000069'
MQCMD_CREATE_BUFFER_POOL	106	X'0000006A'
MQCMD_CREATE_PAGE_SET	107	X'0000006B'
MQCMD_CREATE_CF_STRUC	108	X'0000006C'
MQCMD_CREATE_STG_CLASS	109	X'0000006D'
MQCMD_COPY_CF_STRUC	110	X'0000006E'
CLASSE MQCMD_COPY_STG_CLASS	111	X'0000006F'
MQCMD_DELETE_CF_STRUC	112	X'00000070'
CLASSE MQCMD_DELETE_STG_CLASS	113	X'00000071'
MQCMD_INQUIRE_ARCHIVE	114	X'00000072'
MQCMD_INQUIRE_CF_STRUC	115	X'00000073'
MQCMD_INQUIRE_CF_STRUC_STATUS	116	X'00000074'
MQCMD_INQUIRE_CMD_SERVER	117	X'00000075'
MQCMD_INQUIRE_CHANNEL_INIT	118	X'00000076'
MQCMD_INQUIRE_QSG	119	X'00000077'
MQCMD_INQUIRE_LOG	120	X'00000078'
MQCMD_INQUIRE_SECURITY	121	X'00000079'
MQCMD_INQUIRE_STG_CLASS	122	X'0000007A'
MQCMD_INQUIRE_SYSTEM	123	X'0000007B'
MQCMD_INQUIRE_THREAD	124	X'0000007C'
MQCMD_INQUIRE_TRACE	125	X'0000007D'

Tableau 102. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQCMD_INQUIRE_USAGE	126	X'0000007E'
MQCMD_MOVE_Q	127	X'0000007F'
MQCMD_RECOVER_BSDS	128	X'00000080'
MQCMD_RECOVER_CF_STRUC	129	X'00000081'
MQCMD_RESET_TPIPE	130	X'00000082'
MQCMD_RESOLVE_INDOUBT	131	X'00000083'
MQCMD_RESUME_Q_MGR	132	X'00000084'
MQCMD_REVERIFY_SECURITY	133	X'00000085'
MQCMD_SET_ARCHIVE	134	X'00000086'
MQCMD_SET_LOG	136	X'00000088'
MQCMD_SET_SYSTEM	137	X'00000089'
MQCMD_START_CMD_SERVEUR	138	X'0000008A'
MQCMD_START_Q_MGR	139	X'0000008B'
MQCMD_START_TRACE	140	X'0000008C'
MQCMD_STOP_CHANNEL_INIT	141	X'0000008D'
MQCMD_STOP_CHANNEL_LISTENER	142	X'0000008E'
MQCMD_STOP_CMD_SERVER	143	X'0000008F'
MQCMD_STOP_Q_MGR	144	X'00000090'
MQCMD_STOP_TRACE	145	X'00000091'
MQCMD_SUSPEND_Q_MGR	146	X'00000092'
MQCMD_INQUIRE_CF_STRUC_NAMES	147	X'00000093'
MQCMD_INQUIRE_STG_CLASS_NAMES	148	X'00000094'
MQCMD_CHANGE_SERVICE	149	X'00000095'
SERVICE MQCMD_COPY_SERVICE	150	X'00000096'
MQCMD_CREATE_SERVICE	151	X'00000097'
MQCMD_DELETE_SERVICE	152	X'00000098'
MQCMD_INQUIRE_SERVICE	153	X'00000099'
MQCMD_INQUIRE_SERVICE_STATUS	154	X'0000009A'
MQCMD_START_SERVICE	155	X'0000009B'
SERVICE MQCMD_STOP_SERVICE	156	X'0000009C'
MQCMD_DELETE_BUFFER_POOL	157	X'0000009D'
MQCMD_DELETE_PAGE_SET	158	X'0000009E'
MQCMD_CHANGE_BUFFER_POOL	159	X'0000009F'
MQCMD_CHANGE_PAGE_SET	160	X'000000A0'
MQCMD_INQUIRE_Q_MGR_STATUS	161	X'000000A1'
MQCMD_CREATE_LOG	162	X'000000A2'
MQCMD_STATISTICS_MQI	164	X'000000A4'
MQCMD_STATISTICS_Q	165	X'000000A5'
MQCMD_STATISTICS_CHANNEL	166	X'000000A6'

Tableau 102. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQCMD_ACCOUNTING_MQI	167	X'000000A7'
MQCMD_ACCOUNTING_Q	168	X'000000A8'
MQCMD_INQUIRE_AUTH_SERVICE	169	X'000000A9'
MQCMD_CHANGE_TOPIC	170	X'000000AA'
MQCMD_COPY_TOPIC	171	X'000000AB'
MQCMD_CREATE_TOPIC	172	X'000000AC'
MQCMD_DELETE_TOPIC	173	X'000000AD'
MQCMD_INQUIRE_TOPIC	174	X'000000AE'
MQCMD_INQUIRE_TOPIC_NAMES	175	X'000000AF'
MQCMD_INQUIRE_ABONNEMENT	176	X'000000B0'
MQCMD_CREATE_ABONNEMENT	177	X'000000B1'
ABONNEMENT_CHANGE_MQCMD_	178	X'000000B2'
MQCMD_DELETE_ABONNEMENT	179	X'000000B3'
MQCMD_ABONNEMENT_COPIE	181	X'000000B5'
MQCMD_INQUIRE_SUB_STATUS	182	X'000000B6'
MQCMD_INQUIRE_TOPIC_STATUS	183	X'000000B7'
MQCMD_CLEAR_TOPIC_CHAINE	184	X'000000B8'
MQCMD_INQUIRE_PUBSUB_STATUS	185	X'000000B9'
CANAL_PURGE_MQCMD_	195	X'000000C3'

### MQCMDI\_\* (Valeurs d'informations de commande de format de commande)

Tableau 103. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCMDI_CMDScope_ACCEPTED	1	X'00000001'
MQCMDI_CMDScope_GENERATED	2	X'00000002'
MQCMDI_CMDScope_COMPLETED	3	X'00000003'
MQCMDI_QSG_DISP_COMPLETED	4	X'00000004'
MQCMDI_COMMAND_ACCEPTÉ	5	X'00000005'
MQCMDI_CLUSTER_REQUEST_QUEUED	6	X'00000006'
MQCMDI_CHANNEL_INIT_STARTED	7	X'00000007'
MQCMDI_RECOVER_STARTED	11	X'0000000B'
MQCMDI_BACKUP_STARTED	12	X'0000000C'
MQCMDI_RECOVER_COMPLETED	13	X'0000000D'
MQCMDI_SEC_TIMER_ZERO	14	X'0000000E'
MQCMDI_REFRESH_CONFIGURATION	16	X'00000010'
MQCMDI_SEC_SIGNOFF_ERREUR	17	X'00000011'
MQCMDI_IMS_BRIDGE_SUSPENDED	18	X'00000012'
MQCMDI_DB2_SUSPENDED	19	X'00000013'
MQCMDI_DB2_OBSOLETE_MSGS	20	X'00000014'

Tableau 103. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQCMDI_SEC_MAJUSCULE	21	X'00000015'
MQCMDI_SEC_MIXEDCASE	22	X'00000016'

## MQCMDL\_\* (Niveaux de commande)

Tableau 104. Noms et valeurs de constante	
Nom	Valeur
MQCMDL_LEVEL_800	800
MQCMDL_LEVEL_801	801
MQCMDL_LEVEL_802	802
MQCMDL_LEVEL_900	900
MQCMDL_LEVEL_901	901
MQCMDL_LEVEL_902	902
MQCMDL_LEVEL_903	903
MQCMDL_LEVEL_904	904
MQCMDL_LEVEL_905	905
MQCMDL_LEVEL_910	910
MQCMDL_LEVEL_912	912
MQCMDL_LEVEL_913	913
MQCMDL_LEVEL_914	914
MQCMDL_LEVEL_915	915
MQCMDL_LEVEL_920	920
MQCMDL_LEVEL_921	921
MQCMDL_LEVEL_922	922
MQCMDL_LEVEL_923	923
MQCMDL_LEVEL_924	924
MQCMDL_LEVEL_925	925
MQCMDL_LEVEL_930	930
MQCMDL_LEVEL_931	931
MQCMDL_LEVEL_932	932

## MQCMHO\_\* (Créer des options et une structure de descripteur de message)

### Créer une structure d'options de descripteur de message

Tableau 105. Structures de constantes	
Nom	Structure
MQCMHO_STRUC_ID	"CMHO"
MQCMHO_STRUC_ID_ARRAY	'C', 'M', 'H', 'O'

**Remarque :** Le symbole – représente un caractère blanc unique.

Tableau 106. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCMHO_VERSION_1	1	X'00000001'
MQCMHO_CURRENT_VERSION	1	X'00000001'

## Options de création de descripteur de message

Tableau 107. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCMHO_VALEUR_PAR_DÉFAUT	0	X'00000000'
MQCMHO_NO_VALIDATION	1	X'00000001'
MQCMHO_VALIDATION	2	X'00000002'
MQCMHO_AUCUN	0	X'00000000'

## MQCNO\_\* (options et structure de connexion)

### Structure des options de connexion

Tableau 108. Structures de constantes	
Nom	Structure
ID_STRUC_MQCNO	"CNO~"
MQCNO_STRUC_ID_ARRAY	'C', 'N', 'O', '~'

**Remarque :** Le symbole ~ représente un caractère blanc unique.

Tableau 109. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCNO_VERSION_1	1	X'00000001'
MQCNO_VERSION_2	2	X'00000002'
MQCNO_VERSION_3	3	X'00000003'
MQCNO_VERSION_4	4	X'00000004'
MQCNO_VERSION_5	5	X'00000005'
MQCNO_CURRENT_VERSION	5	X'00000005'

### Options de connexion

Tableau 110. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCNO_STANDARD_BINDING	0	X'00000000'
MQCNO_FASTPATH_BINDING	1	X'00000001'
MQCNO_SERIALIZE_CONN_TAG_Q_MGR	2	X'00000002'
MQCNO_SERIALIZE_CONN_TAG_QSG	4	X'00000004'
MQCNO_RESTRICT_CONN_TAG_Q_MGR	8	X'00000008'
MQCNO_RESTRICT_CONN_TAG_QSG	16	X'00000010'
MQCNO_HANDLE_SHARE_NONE	32	X'00000020'
MQCNO_HANDLE_SHARE_BLOCK	64	X'00000040'

Tableau 110. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQCNO_HANDLE_SHARE_NO_BLOCK	128	X'00000080'
MQCNO_LIEN_PARTAGE	256	X'00000100'
MQCNO_LIEN_ISOLÉ_LIAISON	512	X'00000200'
MQCNO_LIEN_LOCAL_LOCAL	1024	X'00000400'
LIEN_CLIENT_MQCNO_BINDING	2048	X'00000800'
MQCNO_ACCOUNTING_MQI_ENABLED	4096	X'00001000'
MQCNO_ACCOUNTING_MQI_DISABLED	8192	X'00002000'
MQCNO_ACCOUNTING_Q_ENABLED	16384	X'00004000'
MQCNO_ACCOUNTING_Q_DISABLED	32768	X'00008000'
MQCNO_NO_CONV_SHARING	65536	X'00010000'
MQCNO_TOUS_CONVS_SHARE	262144	X'00040000'
MQCNO_CD_FOR_OUTPUT_ONLY	524288	X'00080000'
MQCNO_USE_CD_SELECTION	1048576	X'00100000'
MQCNO_RECONNECT	16777216	X'01000000'
MQCNO_RECONNECT_AS_DEF	0	X'00000000'
MQCNO_RECONNECT_DISABLED	33554432	X'02000000'
MQCNO_RECONNECT_Q_MGR	67108864	X'04000000'
MQCNO_ACTIVITY_TRACE_ENABLED	134217728	X'08000000'
MQCNO_ACTIVITY_TRACE_DISABLED	268435456	X'10000000'
MQCNO_AUCUN	0	X'00000000'

### MQCO\_\* (options de fermeture)

Tableau 111. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCO_IMMEDIATE	0	X'00000000'
MQCO_AUCUN	0	X'00000000'
MQCO_DELETE	1	X'00000001'
MQCO_DELETE_PURGE	2	X'00000002'
MQCO_KEEP_SUB	4	X'00000004'
SOUS-TITRE_REMOVE_MQQUE	8	X'00000008'
MQCO QUIESCE	32	X'00000020'

### MQCODL\_\* (longueur des données de sortie de l'en-tête d'information CICS)

Tableau 112. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCODL_AS_ENTREE	-1	X'FFFFFFFF'



## MQCOMPRESS\_\* (compression de canal)

Nom	Valeur décimale	Valeur hexadécimale
MQCOMPRESS_NON_DISPONIBLE	-1	X'FFFFFFFF'
MQCOMPRESS_NONE	0	X'00000000'
MQCOMPRESS_RLE	1	X'00000001'
MQCOMPRESS_ZLIBFAST	2	X'00000002'
MQCOMPRESS_ZLIBHIGH	4	X'00000004'
MQCOMPRESS_SYSTEM	8	X'00000008'
MQCOMPRESS_ANY	268435455	X'0FFFFFFF'

## MQCONNID\_\* (identificateur de connexion)

Nom	Valeur
MQCONNID_NONE	X'00...00' (24 valeurs nulles)
MQCONNID_NONE_ARRAY	'\0', '\0', ... (24 valeurs nulles)

## MQCOPY\_\* (Options de copie de propriété)

Nom	Valeur décimale	Valeur hexadécimale
MQCOPY_NONE	0	X'00000000'
MQCOPY_ALL	1	X'00000001'
MQCOPY_FORWARD	2	X'00000002'
MQCOPY_PUBLICATION	4	X'00000004'
REPONSE_MQCOPY_RÉPONSE	8	X'00000008'
RAPPORT MQCOPY_REPORT	16	X'00000010'
MQCOPY_DEFAULT	22	X'00000016'

## MQCQT\_\* (types de file d'attente de cluster)

Nom	Valeur décimale	Valeur hexadécimale
MQCQT_LOCAL_Q	1	X'00000001'
MQCQT_ALIAS_Q	2	X'00000002'
MQCQT_REMOTE_Q	3	X'00000003'
MQCQT_Q_MGR_ALIAS	4	X'00000004'

## MQCRC\_\* (codes retour de l'en-tête d'information CICS)

Nom	Valeur décimale	Valeur hexadécimale
MQCRC_OK	0	X'00000000'
MQCRC_CICS_EXEC_ERROR	1	X'00000001'

Tableau 117. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQCRC_MQ_API_ERREUR	2	X'00000002'
MQCRC_BRIDGE_ERREUR	3	X'00000003'
MQCRC_BRIDGE_ABEND	4	X'00000004'
MQCRC_APPLICATION_ABEND	5	X'00000005'
MQCRC_SECURITY_ERROR	6	X'00000006'
MQCRC_PROGRAM_NOT_AVAILABLE	7	X'00000007'
MQCRC_DÉLAI_BRIDGE_ATTENTE	8	X'00000008'
MQCRC_TRANSID_NON DISPONIBLE	9	X'00000009'

### MQCS\_\* (constantes MQCBC-Etat du consommateur)

Tableau 118. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCS_AUCUN	0	X'00000000'
MQCS_SUSPENDU_TEMPORAIRE	1	X'00000001'
ACTION MQCS_SUSPENDED_USER_ACTION	2	X'00000002'
MQCS_SUSPENDU	3	X'00000003'
MQCS_ARRETÉE	4	X'00000004'

### MQCSC\_\* (codes de début d'en-tête d'information CICS)

Tableau 119. Structures de constantes	
Nom	Structure
MQCSC_DEBUT	"S¬¬"
Données MQCSC_STARTDATA	"SD¬¬"
PUT MQCSC_TERMINE	"TD¬¬"
MQCSC_NONE	"¬¬¬"
MQCSC_START_ARRAY	'S', '¬', '¬', '¬'
MQCSC_STARTDATA_ARRAY	'S', 'D', '¬', '¬'
MQCSC_TERMINPUT_ARRAY	'T', 'D', '¬', '¬'
MQCSC_NONE_ARRAY	'¬', '¬', '¬', '¬'

**Remarque :** Le symbole ¬ représente un caractère blanc unique.

### MQCSP\_\* (structure des paramètres de sécurité de connexion et types d'authentification)

#### Structure des paramètres de sécurité de connexion

Tableau 120. Structures de constantes	
Nom	Structure
MQCSP_STRUC_ID	"CSP¬"
MQCSP_STRUC_ID_ARRAY	'C', 'S', 'P', '¬'

**Remarque :** Le symbole – représente un caractère blanc unique.

Tableau 121. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCSP_VERSION_1	1	X'00000001'
MQCSP_VERSION_2	2	X'00000002'
MQCSP_VERSION_3	3	X'00000003'
MQCSP_CURRENT_VERSION	3	X'00000003'

### Paramètres de sécurité de connexion Types d'authentification

Tableau 122. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCSP_AUTH_NONE	0	X'00000000'
MQCSP_AUTH_USER_ID_AND_PWD	1	X'00000001'
MQCSP_AUTH_ID_TOKEN	2	X'00000002'

### MQCSRV\_\* (Options du serveur de commandes)

Tableau 123. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCSRV_CONVERT_NO	0	X'00000000'
MQCSRV_CONVERT_YES	1	X'00000001'
MQCSRV_DLQ_NO	0	X'00000000'
MQCSRV_DLQ_YES	1	X'00000001'

### MQCT\_\* (balise de connexion du gestionnaire de files d'attente)

Tableau 124. Noms et valeurs de constante

Nom	Valeur
MQCT_AUCUN	X'00...00' (128 valeurs nulles)
MQCT_NON_ARRAY	'\0', '\0', ... (128 valeurs nulles)

### MQCTES\_\* (en-tête d'information CICS Task End Status)

Tableau 125. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCTES_NOSYNC	0	X'00000000'
MQCTES_COMMIT	256	X'00000100'
MQCTES_BACKOUT	4352	X'00001100'
MQCTES_ENDTASK	65536	X'00010000'

## MQCTLO\_\* (structure des options MQCTL et options de contrôle de consommateur)

### Structure des options MQCTL

Tableau 126. Structures de constantes

Nom	Structure
MQCTLO_STRUC_ID	"CTLO"
MQCTLO_STRUC_ID_ARRAY	'C', 'T', 'L', 'O'

**Remarque :** Le symbole – représente un caractère blanc unique.

Tableau 127. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCTLO_VERSION_1	1	X'00000001'
MQCTLO_CURRENT_VERSION	1	X'00000001'

### Options MQCTL-Options de contrôle du consommateur

Tableau 128. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCTLO_NONE	0	X'00000000'
MQCTLO_AFFINITÉ_UNITÉS d'exécution	1	X'00000001'
MQCTLO_FAIL_IF QUIESCING	8192	X'00002000'

## MQCUOWC\_\* (commandes d'unité de travail de l'en-tête d'informations CICS)

Tableau 129. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCUOWC_ONLY	273	X'00000111'
MQCUOWC_CONTINUÉ	65536	X'00010000'
MQCUOWC_FIRST	17	X'00000011'
MQCUOWC_MILIEU	16	X'00000010'
MQCUOWC_LAST	272	X'00000110'
MQCUOWC_COMMIT	256	X'00000100'
MQCUOWC_BACKOUT	4352	X'00001100'

## MQCXP\_\* (structure des paramètres d'exit de canal)

Tableau 130. Structures de constantes

Nom	Structure
MQCXP_STRUC_ID	"CXP–"
MQCXP_STRUC_ID_ARRAY	'C', 'X', 'P', '–'

**Remarque :** Le symbole – représente un caractère blanc unique.

Tableau 131. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQCXP_VERSION_1	1	X'00000001'
MQCXP_VERSION_2	2	X'00000002'
MQCXP_VERSION_3	3	X'00000003'
MQCXP_VERSION_4	4	X'00000004'
MQCXP_VERSION_5	5	X'00000005'
MQCXP_VERSION_6	6	X'00000006'
MQCXP_VERSION_7	7	X'00000007'
MQCXP_VERSION_8	8	X'00000008'
MQCXP_VERSION_9	9	X'00000009'
MQCXP_CURRENT_VERSION	9	X'00000009'

### MQDC\_\* (classe de destination)

Tableau 132. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQDC_GERE	1	X'00000001'
MQDC_FOURNI	2	X'00000002'

### MQDCC\_\* (options de conversion, masques et facteurs)

#### Options de conversion

Tableau 133. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQDCC_CONVERSION_PAR_DÉFAUT_PAR_DÉFAUT	1	X'00000001'
MQDCC_FILL_TARGET_BUFFER	2	X'00000002'
MQDCC_INT_DEFAULT_CONVERSION	4	X'00000004'
MQDCC_SOURCE_ENC_NATIVE	(value differs by platform or version)	(value differs by platform or version)
MQDCC_SOURCE_ENC_NORMAL	16	X'00000010'
MQDCC_SOURCE_ENC_INVERSE	32	X'00000020'
MQDCC_SOURCE_ENC_UNDEFINED	0	X'00000000'
MQDCC_TARGET_ENC_NATIVE	(value differs by platform or version)	(value differs by platform or version)
MQDCC_CIBLE_ENC_NORMAL	256	X'00000100'
MQDCC_CIBLE_INVERSÉ	512	X'00000200'
MQDCC_TARGET_ENC_UNDEFINED	0	X'00000000'
MQDCC_AUCUN	0	X'00000000'

#### Masques et facteurs des options de conversion

Tableau 134. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MASQUE_SOURCE_MQDCC_CACHE	240	X'000000F0'

Tableau 134. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQDCC_MASQUE_CIBLE_CACHE	3840	X'00000F00'
FACTEUR_SOURCE_MQDCC_SOURCE	16	X'00000010'
MQDCC_FACTEUR_CIBLE	256	X'00000100'

### MQDELO\_\* (Options de suppression de publication / abonnement)

Tableau 135. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQDELO_AUCUN	0	X'00000000'
MQDELO_LOCAL	4	X'00000004'

### MQDH\_\* (Structure d'en-tête de distribution)

Tableau 136. Structures de constantes	
Nom	Structure
ID_STRUC_MQDH_	"DH↯"
MQDH_STRUC_ID_ARRAY	'D','H','↯','↯'

**Remarque :** Le symbole ↯ représente un caractère blanc unique.

Tableau 137. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQDH_VERSION_1	1	X'00000001'
MQDH_CURRENT_VERSION	1	X'00000001'

### MQDHF\_\* (indicateurs d'en-tête de distribution)

Tableau 138. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQDHF_NEW_MSG_IDS	1	X'00000001'
MQDHF_NONE	0	X'00000000'

### MQDISCONNECT\_\* (types de déconnexion du format de commande)

Tableau 139. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQDISCONNECT_NORMAL	0	X'00000000'
MQDISCONNECT_IMPLICITE	1	X'00000001'
MQDISCONNECT_Q_MGR	2	X'00000002'

### MQDL\_\* (Listes de distribution)

Tableau 140. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQDL_SUPPORTED	1	X'00000001'
MQDL_NOT_SUPPORTED	0	X'00000000'

## MQDLH\_\* (structure d'en-tête de rebut)

Tableau 141. Structures de constantes	
Nom	Structure
ID_STRUC_MQDLH	"DLH↵"
MQDLH_STRUC_ID_ARRAY	'D', 'L', 'H', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

MQDLH_VERSION_1	1	X'00000001'
MQDLH_CURRENT_VERSION	1	X'00000001'

## MQDLV\_\* (Distribution des messages persistants / non persistants)

Tableau 142. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQDLV_AS_PARENT	0	X'00000000'
MQDLV_ALL	1	X'00000001'
MQDLV_ALL_DUR	2	X'00000002'
MQDLV_ALL_AVAIL	3	X'00000003'

## MQDMHO\_\* (Supprimer les options et la structure de descripteur de message)

### Supprimer la structure des options de descripteur de message

Tableau 143. Structures de constantes	
Nom	Structure
MQDMHO_STRUC_ID	"DMHO"
MQDMHO_STRUC_ID_ARRAY	'D', 'M', 'H', 'O'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 144. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQDMHO_VERSION_1	1	X'00000001'
MQDMHO_CURRENT_VERSION	1	X'00000001'

### Options de suppression de descripteur de message

Tableau 145. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQDMHO_AUCUN	0	X'00000000'

## MQDMPO\_\* (suppression des options et de la structure des propriétés de message)

### Supprimer la structure des options de propriété de message

Nom	Structure
ID_STRUC_MQDMPO_	"DMPO"
MQDMPO_STRUC_ID_ARRAY	'D', 'M', 'P', 'O'

**Remarque :** Le symbole – représente un caractère blanc unique.

Nom	Valeur décimale	Valeur hexadécimale
MQDMPO_VERSION_1	1	X'00000001'
MQDMPO_VERSION	1	X'00000001'

### Options de suppression de propriété de message

Nom	Valeur décimale	Valeur hexadécimale
MQDMPO_DEL_FIRST	0	X'00000000'
MQDMPO_DEL_PROP_UNDER_CURSOR	1	X'00000001'
MQDMPO_AUCUN	0	X'00000000'

## MQDNSWLM\_\* (WLM DNS)

Nom	Valeur décimale	Valeur hexadécimale
MQDNSWLM_NO	0	X'00000000'
MQDNSWLM_OUI	1	X'00000001'

## MQDT\_\* (types de destination)

Nom	Valeur décimale	Valeur hexadécimale
MQDT_APPL	1	X'00000001'
COURTIER MQDT_BROKER	2	X'00000002'

## MQDXP\_\* (structure des paramètres d'exit de conversion)

Nom	Structure
ID_STRUC_MQDXP	"DXP–"
MQDXP_STRUC_ID_ARRAY	'D', 'X', 'P', '–'

**Remarque :** Le symbole – représente un caractère blanc unique.



Tableau 152. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQDXP_VERSION_1	1	X'00000001'
MQDXP_VERSION_2	2	X'00000002'
MQDXP_CURRENT_VERSION	2	X'00000002'

### MQEC\_\* (Valeurs de signal)

Tableau 153. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQEC_MSG_ARRIVÉ	2	X'00000002'
MQEC_INTERVALLE_ATTENTE_EXPIRÉE	3	X'00000003'
MQEC_ATTENTE_ANNULE	4	X'00000004'
MQEC_Q_MGR QUIESCING	5	X'00000005'
MQEC_CONNEXION_MISE au repos	6	X'00000006'

### MQEI\_\* (expiration)

Tableau 154. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQEI_UNLIMITED	-1	X'FFFFFFFF'

### MQENC\_\* (codage)

### MQENC\_\* (codage)

Tableau 155. Valeurs des constantes par plateforme			
Nom	Plateforme	Valeur décimale	Valeur hexadécimale
MQENC_NATIVE	IBM i	273	X'00000111'
	Linux	546	X'00000222'
	Linux sur SPARC	273	X'00000111'
	Linux sur x86	546	X'00000222'
	AIX and Linux	273	X'00000111'
	Windows	546	X'00000222'
	Micro-Mise en évidence de COBOL sur Windows	17	X'00000011'
	z/OS	785	X'00000311'

### MQENC\_\* (masques de codage)

Tableau 156. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQENC_INTEGER_MASK	15	X'0000000F'
MQENC_DECIMAL_MASK	240	X'000000F0'
MQENC_FLOAT_MASK	3840	X'00000F00'
MASQUE_RÉSERVÉ_MQENC_MASQUE	-4096	X'FFFFFF00'

## MQENC\_\* (Encodages pour les entiers binaires)

Tableau 157. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQENC_INTEGER_UNDEFINI	0	X'00000000'
MQENC_INTEGER_NORMAL	1	X'00000001'
MQENC_INTEGER_REVERSED	2	X'00000002'

## MQENC\_\* (Codages pour les entiers décimaux condensés)

Tableau 158. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQENC_DECIMAL_UNDEFINI	0	X'00000000'
MQENC_DECIMAL_NORMAL	16	X'00000010'
MQENC_DECIMAL_REVERSED	32	X'00000020'

## MQENC\_\* (Encodages pour les nombres à virgule flottante)

Tableau 159. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQENC_FLOAT_NON_DEFINI	0	X'00000000'
MQENC_FLOAT_IEEE_NORMAL	256	X'00000100'
MQENC_FLOAT_IEEE_REVERSED	512	X'00000200'
MQENC_FLOAT_S390	768	X'00000300'
MQENC_FLOAT_TNS	1024	X'00000400'

## MQEPH\_\* (indicateurs et structure d'en-tête de format de commande imbriquée)

### Structure d'en-tête de format de commande intégrée

Tableau 160. Structures de constantes	
Nom	Structure
ID_STRUC_MQEPH_ID	"EPH↵"
MQEPH_STRUC_ID_ARRAY	'E', 'P', 'H', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 161. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQEPH_STRUC_LENGTH_FIXED	68	X'00000044'
MQEPH_VERSION_1	1	X'00000001'
MQEPH_CURRENT_VERSION	1	X'00000001'

## Indicateurs d'en-tête de format de commande imbriquée

Tableau 162. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQEPH_NONE	0	X'00000000'
MQEPH_CCSID_IMBRIQUÉ	1	X'00000001'

## MQET\_\* (types d'échappement de format de commande)

Tableau 163. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQET_MQSC	1	X'00000001'

## MQEVO\_\* (origines d'événement de format de commande)

Tableau 164. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQEVO_AUTRES	0	X'00000000'
MQEVO_CONSOLE	1	X'00000001'
MQEVO_INIT	2	X'00000002'
MSG MQEVO_MQ	3	X'00000003'
MQEVO_MQSET	4	X'00000004'
MQEVO_INTERNE	5	X'00000005'
MQEVO_MQSUB	6	X'00000006'
MQEVO_CTLMSG	7	X'00000007'
MQEVO_REST	8	X'00000008'

## MQEVR\_\* (format de commande-Enregistrement d'événements)

Tableau 165. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQEVR_DISABLED	0	X'00000000'
MQEVR_ENABLED	1	X'00000001'
MQEVR_EXCEPTION	2	X'00000002'
MQEVR_NO_DISPLAY	3	X'00000003'

## MQEXPI\_\* (Intervalle d'analyse de l'expiration)

Tableau 166. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQEXPI_OFF	0	X'00000000'

## MQFB\_\* (valeurs de commentaires en retour)

Tableau 167. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQFB_AUCUN	0	X'00000000'

Tableau 167. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQFB_SYSTEM_FIRST	1	X'00000001'
MQFB_QUIT	256	X'00000100'
EXPIRATION_MQFB	258	X'00000102'
MQFB_COA	259	X'00000103'
MQFB_COD	260	X'00000104'
MQFB_CHANNEL_COMPLETED	262	X'00000106'
MQFB_CHANNEL_FAIL_RETRY	263	X'00000107'
MQFB_CHANNEL_FAIL	264	X'00000108'
MQFB_APPL_CANNOT_BE_STARTED	265	X'00000109'
MQFB_TM_ERREUR	266	X'0000010A'
MQFB_APPL_TYPE_ERREUR	267	X'0000010B'
MQFB_STOPPED_BY_MSG_EXIT	268	X'0000010C'
ACTIVITE MQFB_MQ	269	X'0000010D'
MQFB_XMIT_Q_MSG_ERROR	271	X'0000010F'
MQFB_PAN	275	X'00000113'
MQFB_NAN	276	X'00000114'
MQFB_STOPPED_BY_CHAD_EXIT	277	X'00000115'
MQFB_STOPPED_BY_PUBSUB_EXIT	279	X'00000117'
MQFB_NOT_REPOSITORY_MSG	280	X'00000118'
MQFB_BIND_OPEN_CLUSRCVR_DEL	281	X'00000119'
ACTIVITÉS MQFB_MAX_ACTIVITÉS	282	X'0000011A'
MQFB_NON_RÉACHEMINEMENT	283	X'0000011B'
MQFB_NOT_DELIVERED	284	X'0000011C'
MQFB_UNUPPORT_FORWARDING	285	X'0000011D'
MQFB_UNUNSUPPORTED_DISTRIBUTION	286	X'0000011E'
MQFB_DATA_LENGTH_ZERO	291	X'00000123'
MQFB_DONNÉES_LONGUEUR_NÉGATIVE	292	X'00000124'
MQFB_DATA_LENGTH_TOO_BIG	293	X'00000125'
MQFB_BUFFER_OVERFLOW	294	X'00000126'
MQFB_LENGTH_OFF_BY_ONE	295	X'00000127'
MQFB_IIH_ERREUR	296	X'00000128'
MQFB_NOT_AUTHORIZED_FOR_IMS	298	X'0000012A'
MQFB_IMS_ERREUR	300	X'0000012C'
MQFB_IMS_FIRST	301	X'0000012D'
MQFB_IMS_LAST	399	X'0000018F'
MQFB_CICS_INTERNAL_ERROR	401	X'00000191'
MQFB_CICS_NOT_AUTHORIZED	402	X'00000192'
MQFB_CICS_BRIDGE_FAILURE	403	X'00000193'
MQFB_CICS_CORREL_ID_ERREUR	404	X'00000194'

Tableau 167. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQFB_CICS_CC SID_ERREUR	405	X'00000195'
MQFB_CICS_ENCODING_ERROR	406	X'00000196'
MQFB_CICS_CIH_ERREUR	407	X'00000197'
MQFB_ERREUR UOW_CICS_	408	X'00000198'
MQFB_CICS_COMMAREA_ERREUR	409	X'00000199'
MQFB_CICS_APPL_NOT_STARTED	410	X'0000019A'
MQFB_CICS_APPL_ABENDED	411	X'0000019B'
MQFB_CICS_DLQ_ERREUR	412	X'0000019C'
MQFB_CICS_UOW_BACKED_OUT	413	X'0000019D'
MQFB_PUBLICATIONS_ON_REQUEST	501	X'000001F5'
MQFB_SUBSCRIBER_IS_PUBLISHER	502	X'000001F6'
MQFB_MSG_SCOPE_MISMATCH	503	X'000001F7'
MQFB_SELECTOR_MISMATCH	504	X'000001F8'
MQFB_IMS_NACK_1A_REASON_FIRST	600	X'00000258'
MQFB_IMS_NACK_1A_REASON_LAST	855	X'00000357'
MQFB_SYSTEM_LAST	65535	X'0000FFFF'
MQFB_APPL_FIRST	65536	X'00010000'
MQFB_APPL_LAST	99999999	X'3B9AC9FF'

### MQFC\_\* (Options de force de format de commande)

Tableau 168. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQFC_OUI	1	X'00000001'
MQFC_NO	0	X'00000000'

### MQFMT\_\* (formats)

Tableau 169. Noms et valeurs de constante

Nom	Valeur
MQFMT_AUCUN	"rrrrrrrr"
MQFMT_ADMIN	"MQADMINr"
MQFMT_CHANNEL_COMPLETED	"MQCHCOMr"
MQFMT_CICS	"MQCICSrr"
MQFMT_COMMAND_1	"MQCMD1rr"
MQFMT_COMMAND_2	"MQCMD2rr"
MQFMT_DEAD_LETTER_HEADER	"MQDEADr"
MQFMT_DIST_HEADER	"MQHDISTr"
MQFMT_EMBEDDED_PCF	"MQHEPCFr"
MQFMT_EVENT	"MQEVENTr"
MQFMT_IMS	"MQIMSrrr"

Tableau 169. Noms et valeurs de constante (suite)	
Nom	Valeur
MQFMT_IMS_VAR_STRING	"MQIMSVS↵"
MQFMT_MD_EXTENSION	"MQHMDE↵↵"
MQFMT_PCF	"MQPCF↵↵↵"
MQFMT_REF_MSG_HEADER	"MQHREF↵↵"
MQFMT_RF_HEADER	"MQHRF↵↵↵"
MQFMT_RF_HEADER_1	"MQHRF↵↵↵"
MQFMT_RF_HEADER_2	"MQHRF2↵↵"
MQFMT_STRING	"MQSTR↵↵↵"
MQFMT_TRIGGER	"MQTRIG↵↵"
MQFMT_WORK_INFO_HEADER	"MQHWIH↵↵"
MQFMT_XMIT_Q_HEADER	"MQXMIT↵↵"
MQFMT_NONE_ARRAY	'↵','↵','↵','↵','↵','↵','↵','↵','↵'
MQFMT_ADMIN_ARRAY	'M','Q','A','D','M','I','N','↵'
MQFMT_CHANNEL_COMPLETED_ARRAY	'M','Q','C','H','C','O','M','↵'
MQFMT_CICS_ARRAY	'M','Q','C','I','C','S','↵','↵'
MQFMT_COMMAND_1_ARRAY	'M','Q','C','M','D','1','↵','↵'
MQFMT_COMMAND_2_ARRAY	'M','Q','C','M','D','2','↵','↵'
MQFMT_DEAD_LETTER_HEADER_ARRAY	'M','Q','D','E','A','D','↵','↵'
MQFMT_DIST_HEADER_ARRAY	'M','Q','H','D','I','S','T','↵'
MQFMT_EMBEDDED_PCF_ARRAY	'M','Q','H','E','P','C','F','↵'
MQFMT_EVENT_ARRAY	'M','Q','E','V','E','N','T','↵'
MQFMT_IMS_ARRAY	'M','Q','I','M','S','↵','↵','↵'
MQFMT_IMS_VAR_STRING_ARRAY	'M','Q','I','M','S','V','S','↵'
MQFMT_MD_EXTENSION_ARRAY	'M','Q','H','M','D','E','↵','↵'
MQFMT_PCF_ARRAY	'M','Q','P','C','F','↵','↵','↵'
MQFMT_REF_MSG_HEADER_ARRAY	'M','Q','H','R','E','F','↵','↵'
MQFMT_RF_HEADER_ARRAY	'M','Q','H','R','F','↵','↵','↵'
MQFMT_RF_HEADER_1_ARRAY	'M','Q','H','R','F','↵','↵','↵'
MQFMT_RF_HEADER_2_ARRAY	'M','Q','H','R','F','2','↵','↵'
MQFMT_STRING_ARRAY	'M','Q','S','T','R','↵','↵','↵'
MQFMT_TRIGGER_ARRAY	'M','Q','T','R','I','G','↵','↵'
MQFMT_WORK_INFO_HEADER_ARRAY	'M','Q','H','W','I','H','↵','↵'
MQFMT_XMIT_Q_HEADER_ARRAY	'M','Q','X','M','I','T','↵','↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

### MQFUN\_\* (types de fonction d'application)

Tableau 170. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQFUN_TYPE_INCONNU	0	X'00000000'

Tableau 170. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQFUN_TYPE_JVM	1	X'00000001'
PROGRAMME MQFUN_TYPE_PROGRAM	2	X'00000002'
MQFUN_TYPE_PROCEDURE	3	X'00000003'
MQFUN_TYPE_USERDEF	4	X'00000004'
MQFUN_TYPE_COMMANDE	5	X'00000005'

### MQGA\_\* (Sélecteurs d'attributs de groupe)

Tableau 171. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQGA_FIRST	8001	X'00001F41'
MQGA_LAST	9000	X'00002328'

### MQGACF\_\* (Types de paramètre de groupe de format de commande)

Tableau 172. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQGACF_FIRST	8001	X'00001F41'
CONTEXTE_COMMANDE_MQGACF_COMMANDE	8001	X'00001F41'
DONNÉES_COMMANDE_MQGF_ACF	8002	X'00001F42'
MQGACF_TRACE_ROUTE	8003	X'00001F43'
MQGACF_OPERATION	8004	X'00001F44'
ACTIVITÉ_MQGACF_MQG	8005	X'00001F45'
MQGACF_EMBEDDED_MQMD	8006	X'00001F46'
MQGACF_MESSAGE	8007	X'00001F47'
MQGACF_MQMD	8008	X'00001F48'
MQGACF_NOM_VALEUR_	8009	X'00001F49'
MQGACF_Q_DONNEES_COMPTE	8010	X'00001F4A'
MQGACF_Q_STATISTICS_DATA	8011	X'00001F4B'
MQGACF_CHL_STATISTICS_DATA	8012	X'00001F4C'
MQGACF_LAST_USED	8012	X'00001F4C'

### MQGI\_\* (identificateur de groupe)

Tableau 173. Noms et valeurs de constante

Nom	Valeur
MQGI_AUCUN	X'00...00' (24 valeurs nulles)
MQGI_NON_ARRAY	'\0', '\0', ... (24 valeurs nulles)

## MQGMO\_\* (Obtenir les options et la structure du message)

### Obtenir la structure des options de message

Nom	Structure
ID_STRUC_MQGMO_	"GMO↵"
MQGMO_STRUC_ID_ARRAY	'G', 'M', 'O', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Nom	Valeur décimale	Valeur hexadécimale
MQGMO_VERSION_1	1	X'00000001'
MQGMO_VERSION_2	2	X'00000002'
MQGMO_VERSION_3	3	X'00000003'
MQGMO_VERSION_4	4	X'00000004'
MQGMO_CURRENT_VERSION	4	X'00000004'

### Options d'obtention de message

Nom	Valeur décimale	Valeur hexadécimale
MQGMO_WAIT	1	X'00000001'
MQGMO_NO_WAIT	0	X'00000000'
MQGMO_SET_SIGNAL	8	X'00000008'
MQGMO_FAIL_IF QUIESCING	8192	X'00002000'
MQGMO_SYNCPOINT	2	X'00000002'
MQGMO_SYNCPOINT_IF_PERSISTENT	4096	X'00001000'
MQGMO_NO_SYNCPOINT	4	X'00000004'
MQGMO_MARK_SKIP_BACKOUT	128	X'00000080'
MQGMO_BROWSE_FIRST	16	X'00000010'
MQGMO_BROWSE_NEXT	32	X'00000020'
MQGMO_BROWSE_MSG_UNDER_CURSOR	2048	X'00008000'
MQGMO_BROWSE_HANDLE	17825808	X'01100010'
MQGMO_BROWSE_CO_OP	18874384	X'01200010'
MQGMO_MSG_UNDER_CURSOR	256	X'00000100'
MQGMO_LOCK	512	X'00000200'
MQGMO_UNLOCK	1024	X'00000400'
MQGMO_ACCEPT_TRUNCATED_MSG	64	X'00000040'
MQGMO_CONVERT	16384	X'00004000'
MQGMO_LOGICAL_ORDER	32768	X'00008000'
MQGMO_COMPLETE_MSG	65536	X'00010000'
MQGMO_ALL_MSGS_AVAILABLE	131072	X'00020000'
MQGMO_ALL_SEGMENTS_AVAILABLE	262144	X'00040000'



Tableau 176. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQGMO_MARK_BROWSE_HANDLE	1048576	X'00100000'
MQGMO_MARK_BROWSE_CO_OP	2097152	X'00200000'
MQGMO_UNMARK_BROWSE_CO_OP	4194304	X'00400000'
MQGMO_UNMARK_BROWSE_HANDLE	8388608	X'00800000'
MQGMO_UNMARKED_BROWSE_MSG	16777216	X'01000000'
MQGMO_PROPERTIES_FORCE_MQRFH2	33554432	X'02000000'
MQGMO_NO_PROPERTIES	67108864	X'04000000'
MQGMO_PROPERTIES_IN_HANDLE	134217728	X'08000000'
MQGMO_PROPERTIES_COMPATIBILITY	268435456	X'10000000'
MQGMO_PROPERTIES_AS_Q_DEF	0	X'00000000'
MQGMO_NONE	0	X'00000000'

### MQGS\_\* (statut du groupe)

Tableau 177. Noms et valeurs de constante	
Nom	Valeur
MQGS_NOT_IN_GROUPE	'␣'
MQGS_MSG_IN_GROUPE	'G'
MQGS_LAST_MSG_IN_GROUP	'L'

**Remarque :** Le symbole ␣ représente un caractère blanc unique.

### MQHA\_\* (Sélecteurs de descripteur)

Tableau 178. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQHA_FIRST	4001	X'00000FA1'
MQHA_BAG_HANDLE	4001	X'00000FA1'
MQHA_LAST_USED	4001	X'00000FA1'
MQHA_LAST	6000	X'00001770'

### MQHB\_\* (descripteurs de sac)

Tableau 179. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQHB_UNUSABLE_HBAG	-1	X'FFFFFFFF'
MQHB_AUCUN	-2	X'FFFFFFFE'

### MQHC\_\* (descripteurs de connexion)

Tableau 180. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQHC_DEF_HCONN	0	X'00000000'
MQHC_UNUSABLE_HCONN	-1	X'FFFFFFFF'

Tableau 180. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQHC_UNASSOCIATED_HCONN	-3	X'FFFFFFFD'

### MQHM\_\* (descripteur de message)

Tableau 181. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQHM_UNUSABLE_HMSG	-1	X'FFFFFFF'
MQHM_NONE	0	X'00000000'

### MQHO\_\* (Descripteur d'objet)

Tableau 182. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQHO_UNUSABLE_HOBJ	-1	X'FFFFFFF'
MQHO_AUCUN	0	X'00000000'

### MQHSTATE\_\* (Etats de descripteur de format de commande)

Tableau 183. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQHSTATE_INACTIVE	0	X'00000000'
MQHSTATE_ACTIVE	1	X'00000001'

### MQIA\_\* (Sélecteurs d'attribut d'entier)

Tableau 184. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQIA_ACCOUNTING_CONN_OVERRIDE	136	X'00000088'
MQIA_ACCOUNTING_INTERVAL	135	X'00000087'
MQIA_ACCOUNTING_MQI	133	X'00000085'
MQIA_ACCOUNTING_Q	134	X'00000086'
MQIA_ACTIVE_CHANNELS	100	X'00000064'
MQIA_ACTIVITY_CONN_OVERRIDE	239	X'000000EF'
MQIA_ACTIVITY_RECORDING	138	X'0000008A'
MQIA_ACTIVITY_TRACE	240	X'000000F0'
MQIA_ADOPTNEWMCA_CHECK	102	X'00000066'
MQIA_ADOPTNEWMCA_INTERVAL	104	X'00000068'
MQIA_ADOPTNEWMCA_TYPE	103	X'00000067'
MQIA_ADOPT_CONTEXT	260	X'00000104'
<b>MQ Adv. VUE</b> MQIA_ADVANCED_CAPABILITY	273	X'00000111'
MQIA_AMQP_CAPABILITY	265	X'00000109'
MQIA_APPL_TYPE	1	X'00000001'
MQIA_ARCHIVE	60	X'0000003C'

Tableau 184. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQIA_AUTHENTICATION_FAIL_DELAY	259	X'00000103'
MQIA_AUTHENTICATION_METHOD	266	X'0000010A'
MQIA_AUTH_INFO_TYPE	66	X'00000042'
MQIA_AUTHORITY_EVENT	47	X'0000002F'
MQIA_AUTO_REORG_INTERVAL	174	X'000000AE'
MQIA_AUTO_REORGANIZATION	173	X'000000AD'
MQIA_BACKOUT_THRESHOLD	22	X'00000016'
MQIA_BASE_TYPE	193	X'000000C1'
MQIA_BATCH_INTERFACE_AUTO	86	X'00000056'
MQIA_BRIDGE_EVENT	74	X'0000004A'
MQIA_CF_LEVEL	70	X'00000046'
MQIA_CF_RECOVER	71	X'00000047'
MQIA_CHANNEL_AUTO_DEF	55	X'00000037'
MQIA_CHANNEL_AUTO_DEF_EVENT	56	X'00000038'
MQIA_CHANNEL_EVENT	73	X'00000049'
MQIA_CHECK_CLIENT_BINDING	258	X'00000102'
MQIA_CHECK_LOCAL_BINDING	257	X'00000101'
MQIA_CHINIT_ADAPTERS	101	X'00000065'
MQIA_CHINIT_CONTROL	119	X'00000077'
MQIA_CHINIT_DISPATCHERS	105	X'00000069'
MQIA_CHINIT_TRACE_AUTO_START	117	X'00000075'
MQIA_CHINIT_TRACE_TABLE_SIZE	118	X'00000076'
MQIA_CLUSTER_OBJECT_STATE	256	X'00000100'
MQIA_CLUSTER_PUB_ROUTE	255	X'000000FF'
MQIA_CLUSTER_Q_TYPE	59	X'0000003B'
MQIA_CLUSTER_WORKLOAD_LENGTH	58	X'0000003A'
MQIA_CLWL_MRU_CHANNELS	97	X'00000061'
MQIA_CLWL_Q_RANK	95	X'0000005F'
MQIA_CLWL_Q_PRIORITY	96	X'00000060'
MQIA_CLWL_USEQ	98	X'00000062'
MQIA_CMD_SERVER_AUTO	87	X'00000057'
MQIA_CMD_SERVER_CONTROL	120	X'00000078'
MQIA_CMD_SERVER_CONVERT_MSG	88	X'00000058'
MQIA_CMD_SERVER_DLQ_MSG	89	X'00000059'
MQIA_CODED_CHAR_SET_ID	2	X'00000002'
MQIA_COMM_EVENT	232	X'000000E8'
MQIA_COMMAND_EVENT	99	X'00000063'
MQIA_COMMAND_LEVEL	31	X'0000001F'
MQIA_CONFIGURATION_EVENT	51	X'00000033'

Tableau 184. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQIA_CPI_LEVEL	27	X'0000001B'
MQIA_CURRENT_Q_DEPTH	3	X'00000003'
MQIA_DEF_BIND	61	X'0000003D'
MQIA_DEF_CLUSTER_XMIT_Q_TYPE	250	X'000000FA'
MQIA_DEF_INPUT_OPEN_OPTION	4	X'00000004'
MQIA_DEF_PERSISTENCE	5	X'00000005'
MQIA_DEF_PRIORITY	6	X'00000006'
MQIA_DEF_PUT_RESPONSE_TYPE	184	X'000000B8'
MQIA_DEF_READ_AHEAD	188	X'000000BC'
MQIA_DEFINITION_TYPE	7	X'00000007'
MQIA_DISPLAY_TYPE	262	X'00000106'
MQIA_DIST_LISTS	34	X'00000022'
MQIA_DNS_WLM	106	X'0000006A'
MQIA_DURABLE_SUB	175	X'000000AF'
MQIA_EXPIRY_INTERVAL	39	X'00000027'
MQIA_FIRST	1	X'00000001'
MQIA_GROUP_UR	221	X'000000DD'
MQIA_HARDEN_GET_BACKOUT	8	X'00000008'
MQIA_HIGH_Q_DEPTH	36	X'00000024'
MQIA_IGQ_PUT_AUTHORITY	65	X'00000041'
MQIA_INDEX_TYPE	57	X'00000039'
MQIA_INHIBIT_EVENT	48	X'00000030'
MQIA_INHIBIT_GET	9	X'00000009'
MQIA_INHIBIT_PUB	181	X'000000B5'
MQIA_INHIBIT_PUT	10	X'0000000A'
MQIA_INHIBIT_SUB	182	X'000000B6'
MQIA_INTRA_GROUP_queuing	64	X'00000040'
MQIA_IP_ADDRESS_VERSION	93	X'0000005D'
MQIA_KEY_REUSE_COUNT	267	X'0000010B'
MQIA_LAST	2000	X'000007D0'
MQIA_LAST_USED	267	X'0000010B'
MQIA_LDAP_AUTHORMD	263	X'00000107'
MQIA_LDAP_NESTGRP	264	X'00000108'
MQIA_LDAP_SECURE_COMM	261	X'00000105'
MQIA_LISTENER_PORT_NUMBER	85	X'00000055'
MQIA_LISTENER_TIMER	107	X'0000006B'
MQIA_LOGGER_EVENT	94	X'0000005E'
MQIA_LU62_CHANNELS	108	X'0000006C'
MQIA_LOCAL_EVENT	49	X'00000031'

Tableau 184. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQIA_MSG_MARK_BROWSE_INTERVAL	68	X'00000044'
MQIA_MAX_CHANNELS	109	X'0000006D'
MQIA_MAX_CLIENTS	172	X'000000AC'
MQIA_MAX_GLOBAL_LOCKS	83	X'00000053'
MQIA_MAX_HANDLES	11	X'0000000B'
MQIA_MAX_LOCAL_LOCKS	84	X'00000054'
MQIA_MAX_MSG_LENGTH	13	X'0000000D'
MQIA_MAX_OPEN_Q	80	X'00000050'
MQIA_MAX_PRIORITY	14	X'0000000E'
MQIA_MAX_PROPERTIES_LENGTH	192	X'000000C0'
MQIA_MAX_Q_DEPTH	15	X'0000000F'
MQIA_MAX_Q_TRIGGERS	90	X'0000005A'
MQIA_MAX_RECOVERY_TASKS	171	X'000000AB'
MQIA_MAX_UNCOMMITTED_MSGS	33	X'00000021'
MQIA_MCAST_BRIDGE	233	X'000000E9'
MQIA_MONITOR_INTERVAL	81	X'00000051'
MQIA_MONITORING_AUTO_CLUSSDR	124	X'0000007C'
MQIA_MONITORING_CHANNEL	122	X'0000007A'
MQIA_MONITORING_Q	123	X'0000007B'
MQIA_MSG_DELIVERY_SEQUENCE	16	X'00000010'
MQIA_MSG_DEQ_COUNT	38	X'00000026'
MQIA_MSG_ENQ_COUNT	37	X'00000025'
MQIA_NAME_COUNT	19	X'00000013'
MQIA_NAMELIST_TYPE	72	X'00000048'
MQIA_NPM_CLASS	78	X'0000004E'
MQIA_NPM_DELIVERY	196	X'000000C4'
MQIA_OPEN_INPUT_COUNT	17	X'00000011'
MQIA_OPEN_OUTPUT_COUNT	18	X'00000012'
MQIA_OUTBOUND_PORT_MAX	140	X'0000008C'
MQIA_OUTBOUND_PORT_MIN	110	X'0000006E'
MQIA_PAGESET_ID	62	X'0000003E'
MQIA_PERFORMANCE_EVENT	53	X'00000035'
MQIA_PLATFORM	32	X'00000020'
MQIA_PM_DELIVERY	195	X'000000C3'
MQIA_PROPERTY_CONTROL	190	X'000000BE'
MQIA_PROT_POLICY_CAPABILITY	251	X'000000FB'
MQIA_PROXY_SUB	199	X'000000C7'
MQIA_PUB_COUNT	215	X'000000D7'
MQIA_PUB_SCOPE	219	X'000000DB'

Tableau 184. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQIA_PUBSUB_CLUSTER	249	X'000000F9'
MQIA_PUBSUB_MAXMSG_RETRY_COUNT	206	X'000000CE'
MQIA_PUBSUB_MODE	187	X'000000BB'
MQIA_PUBSUB_NP_MSG	203	X'000000CB'
MQIA_PUBSUB_NP_RESP	205	X'000000CD'
MQIA_PUBSUB_SYNC_PT	207	X'000000CF'
MQIA_Q_DEPTH_HIGH_EVENT	43	X'0000002B'
MQIA_Q_DEPTH_HIGH_LIMIT	40	X'00000028'
MQIA_Q_DEPTH_LOW_EVENT	44	X'0000002C'
MQIA_Q_DEPTH_LOW_LIMIT	41	X'00000029'
MQIA_Q_DEPTH_MAX_EVENT	42	X'0000002A'
MQIA_Q_SERVICE_INTERVAL	54	X'00000036'
MQIA_Q_SERVICE_INTERVAL_EVENT	46	X'0000002E'
MQIA_Q_TYPE	20	X'00000014'
MQIA_Q_USERS	82	X'00000052'
MQIA_QMGR_CFCONLOS	245	X'000000F5'
MQIA_QMOPT_CONS_COMMS_MSGS	155	X'0000009B'
MQIA_QMOPT_CONS_CRITICAL_MSGS	154	X'0000009A'
MQIA_QMOPT_CONS_ERROR_MSGS	153	X'00000099'
MQIA_QMOPT_CONS_INFO_MSGS	151	X'00000097'
MQIA_QMOPT_CONS_REORG_MSGS	156	X'0000009C'
MQIA_QMOPT_CONS_SYSTEM_MSGS	157	X'0000009D'
MQIA_QMOPT_CONS_WARNING_MSGS	152	X'00000098'
MQIA_QMOPT_CSMT_ON_ERROR	150	X'00000096'
MQIA_QMOPT_INTERNAL_DUMP	170	X'000000AA'
MQIA_QMOPT_LOG_COMMS_MSGS	162	X'000000A2'
MQIA_QMOPT_LOG_CRITICAL_MSGS	161	X'000000A1'
MQIA_QMOPT_LOG_ERROR_MSGS	160	X'000000A0'
MQIA_QMOPT_LOG_INFO_MSGS	158	X'0000009E'
MQIA_QMOPT_LOG_REORG_MSGS	163	X'000000A3'
MQIA_QMOPT_LOG_SYSTEM_MSGS	164	X'000000A4'
MQIA_QMOPT_LOG_WARNING_MSGS	159	X'0000009F'
MQIA_QMOPT_TRACE_COMMS	166	X'000000A6'
MQIA_QMOPT_TRACE_CONVERSION	168	X'000000A8'
MQIA_QMOPT_TRACE_REORG	167	X'000000A7'
MQIA_QMOPT_TRACE_MQI_CALLS	165	X'000000A5'
MQIA_QMOPT_TRACE_SYSTEM	169	X'000000A9'
MQIA_QSG_DISP	63	X'0000003F'
MQIA_READ_AHEAD	189	X'000000BD'

Tableau 184. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQIA_RECEIVE_TIMEOUT	111	X'0000006F'
MQIA_RECEIVE_TIMEOUT_MIN	113	X'00000071'
MQIA_RECEIVE_TIMEOUT_TYPE	112	X'00000070'
MQIA_REMOTE_EVENT	50	X'00000032'
MQIA_RETENTION_INTERVAL	21	X'00000015'
MQIA_REVERSE_DNS_LOOKUP	254	X'000000FE'
MQIA_SCOPE	45	X'0000002D'
MQIA_SECURITY_CASE	141	X'0000008D'
MQIA_SERVICE_CONTROL	139	X'0000008B'
MQIA_SERVICE_TYPE	121	X'00000079'
MQIA_SHAREABILITY	23	X'00000017'
MQIA_SHARED_Q_Q_MGR_NAME	77	X'0000004D'
MQIA_SSL_EVENT	75	X'0000004B'
MQIA_SSL_FIPS_REQUIRED	92	X'0000005C'
MQIA_SSL_RESET_COUNT	76	X'0000004C'
MQIA_SSL_TASKS	69	X'00000045'
MQIA_START_STOP_EVENT	52	X'00000034'
MQIA_STATISTICS_CHANNEL	129	X'00000081'
MQIA_STATISTICS_AUTO_CLUSSDR	130	X'00000082'
MQIA_STATISTICS_INTERVAL	131	X'00000083'
MQIA_STATISTICS_MQI	127	X'0000007F'
MQIA_STATISTICS_Q	128	X'00000080'
MQIA_SUB_COUNT	204	X'000000CC'
MQIA_SUB_SCOPE	218	X'000000DA'
MQIA_SYNCPOINT	30	X'0000001E'
MQIA_TCP_CHANNELS	114	X'00000072'
MQIA_TCP_KEEP_ALIVE	115	X'00000073'
MQIA_TCP_STACK_TYPE	116	X'00000074'
MQIA_TIME_SINCE_RESET	35	X'00000023'
MQIA_TOPIC_DEF_PERSISTENCE	185	X'000000B9'
MQIA_TOPIC_NODE_COUNT	253	X'000000FD'
MQIA_TOPIC_TYPE	208	X'000000D0'
MQIA_TRACE_ROUTE_RECORDING	137	X'00000089'
MQIA_TREE_LIFE_TIME	183	X'000000B7'
MQIA_TRIGGER_CONTROL	24	X'00000018'
MQIA_TRIGGER_DEPTH	29	X'0000001D'
MQIA_TRIGGER_INTERVAL	25	X'00000019'
MQIA_TRIGGER_MSG_PRIORITY	26	X'0000001A'
MQIA_TRIGGER_TYPE	28	X'0000001C'

Tableau 184. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQIA_TRIGGER_RESTART	91	X'0000005B'
MQIA_USAGE	12	X'0000000C'
MQIA_USE_DEAD_LETTER_Q	234	X'000000EA'
MQIA_USER_LIST	2000	X'000007D0'
MQIA_WILDCARD_OPERATION	216	X'000000D8'
MQIA_XR_CAPABILITY	243	X'000000F3'

### MQIACF\_\* (format de commande-Types de paramètre entier)

Tableau 185. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQIACF_FIRST	1001	X'000003E9'
MQIACF_Q_MGR_ATTRS	1001	X'000003E9'
MQIACF_Q_ATTRS	1002	X'000003EA'
MQIACF_PROCESS_ATTRS	1003	X'000003EB'
MQIACF_NAMELIST_ATTRS	1004	X'000003EC'
MQIACF_FORCE	1005	X'000003ED'
REPLACEMENT_MQIACF_ACF	1006	X'000003EE'
PURGE_MQIACF_ACF	1007	X'000003EF'
MQIACF QUIESCE	1008	X'000003F0'
MODE MQIACF	1008	X'000003F0'
MQIACF_ALL	1009	X'000003F1'
MQIACF_EVENT_APPL_TYPE	1010	X'000003F2'
ORIGINE_ÉVÉNEMENT_MQIACF_ACF	1011	X'000003F3'
ID_PARAMÈTRE_MQIACF_ID	1012	X'000003F4'
ID_ERREUR_MQIACF	1013	X'000003F5'
IDENTIFICATEUR_ERREUR_MQIACF_XX_ENCODE_CASE_CAPS_LOCK_OFF	1013	X'000003F5'
MQIACF_SELECTOR	1014	X'000003F6'
MQIACF_ATTRIBUTES de canal	1015	X'000003F7'
TYPE_OBS_MQIACF	1016	X'000003F8'
MQIACF_TYPE_ÉCHAPPEMENT	1017	X'000003F9'
MQIACF_DÉCALAGE_ERREUR_ERREUR	1018	X'000003FA'
MQIACF_AUTH_INFO_ATTRS	1019	X'000003FB'
QUALIFICATION_RAISON_MQIACF_ANOMALIE	1020	X'000003FC'
Commande MQIACF_COMMAND	1021	X'000003FD'
OPTIONS_OPEN_MQIACF_ACF	1022	X'000003FE'
MQIACF_OPEN_TYPE	1023	X'000003FF'
ID_PROCESSUS_MQIACF_ID	1024	X'00000400'
ID_UNITÉ_UNITÉ_FILE_MQIACF	1025	X'00000401'
MQIACF_Q_STATUS_ATTRS	1026	X'00000402'



Tableau 185. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQIACF_UNCOMMITTED_MSGS	1027	X'00000403'
ETAT_HANDLE_MQIACF	1028	X'00000404'
MQIACF_AUX_ERROR_DATA_INT_1	1070	X'0000042E'
MQIACF_AUX_ERROR_DATA_INT_2	1071	X'0000042F'
MQIACF_CONV_CODE_RAISON	1072	X'00000430'
TYPE_BRIDGE_MQIACF_	1073	X'00000431'
MQIACF_INTERROGATION	1074	X'00000432'
INTERVALLE_ATTENTE_MQIACF_	1075	X'00000433'
OPTIONS MQIACF	1076	X'00000434'
MQIACF_BROKER_OPTIONS	1077	X'00000435'
MQIACF_REFRESH_TYPE	1078	X'00000436'
NUMÉRO_SÉQUENCE_MQIACF_XX_ENCODE_CASE_CAPS_LOCK_OF F	1079	X'00000437'
Données_INTEGER_MQIACF_	1080	X'00000438'
MQIACF_OPTIONS d'enregistrement	1081	X'00000439'
MQIACF_PUBLICATION_OPTIONS	1082	X'0000043A'
MQIACF_CLUSTER_INFO	1083	X'0000043B'
MQIACF_Q_MGR_DEFINITION_TYPE	1084	X'0000043C'
MQIACF_Q_MGR_TYPE	1085	X'0000043D'
ACTION MQIACF	1086	X'0000043E'
MQIACF_INTERRUPTON	1087	X'0000043F'
MQIACF_NOMBRE_COURTIERS	1088	X'00000440'
NOMBRE_APPL_MQIACF_NOMBRE	1089	X'00000441'
NOMBRE_ANONYMISES_MQIACF_XX_ENCODE_CASE_CAPS_LOCK _OFF	1090	X'00000442'
OPTIONS REG_REG_MQIACF	1091	X'00000443'
OPTIONS DE DELETE_MQIACF	1092	X'00000444'
MQIACF_CLUSTER_Q_MGR_ATTRS	1093	X'00000445'
INTERVALLE_ACTUALISATION_MQIACF_	1094	X'00000446'
MQIACF_REFRESH_REPOSITORY	1095	X'00000447'
MQIACF_FILES d'attente de suppression	1096	X'00000448'
MQIACF_TYPE_ENTRÉE_OUVERTE	1098	X'0000044A'
MQIACF_OPEN_OUTPUT	1099	X'0000044B'
MQIACF_OPEN_SET	1100	X'0000044C'
MQIACF_OPEN_INQUIRE	1101	X'0000044D'
MQIACF_OPEN_BROWSE	1102	X'0000044E'
MQIACF_Q_STATUS_TYPE	1103	X'0000044F'
MQIACF_Q_DESCRIPTEUR	1104	X'00000450'
STATUT Q_MQIACF	1105	X'00000451'
TYPE_SÉCURITÉ_MQIACF_XX_ENCODE_CASE_CAPS_LOCK_OFF	1106	X'00000452'

Tableau 185. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQIACF_CONNECTION_ATTRS	1107	X'00000453'
MQIACF_CONNECT_OPTIONS	1108	X'00000454'
MQIACF_CONN_INFO_TYPE	1110	X'00000456'
MQIACF_CONN_INFO_CONN	1111	X'00000457'
MQIACF_CONN_INFO_HANDLE	1112	X'00000458'
MQIACF_CONN_INFO_ALL	1113	X'00000459'
MQIACF_AUTH_PROFIL_ATTRS	1114	X'0000045A'
LISTE_AUTORISATION_MQIACF_AUTORISATION	1115	X'0000045B'
MQIACF_AUTH_ADD_AUTHS	1116	X'0000045C'
MQIACF_AUTH_REMOVE_AUTHS	1117	X'0000045D'
TY_ENTIER_MQIACF_TYPE	1118	X'0000045E'
MQIACF_COMMAND_INFO	1120	X'00000460'
MQIACF_CMDSCOPE_Q_MGR_COUNT	1121	X'00000461'
MQIACF_Q_MGR_SYSTEM	1122	X'00000462'
MQIACF_Q_MGR_EVENT	1123	X'00000463'
MQIACF_Q_MGR_DQM	1124	X'00000464'
MQIACF_Q_MGR_CLUSTER	1125	X'00000465'
MQIACF_QSG_DISPS	1126	X'00000466'
ETAT_UOW_MQIACF	1128	X'00000468'
MQIACF_SECURITY_ITEM	1129	X'00000469'
MQIACF_CF_STATUT_STRUCTURE	1130	X'0000046A'
TYPE_UOW_MQIACF_MQ	1132	X'0000046C'
MQIACF_CF_STRUC_ATTRS	1133	X'0000046D'
MQIACF_INTERVALDE_EXCLUSION	1134	X'0000046E'
MQIACF_CF_STATUS_TYPE	1135	X'0000046F'
MQIACF_CF_STATUS_SUMMARY	1136	X'00000470'
MQIACF_CF_STATUS_CONNECT	1137	X'00000471'
MQIACF_CF_STATUS_BACKUP	1138	X'00000472'
MQIACF_CF_TYPE_STRUCTURE	1139	X'00000473'
MQIACF_CF_STRUC_SIZE_MAX	1140	X'00000474'
MQIACF_CF_STRUC_SIZE_USED	1141	X'00000475'
MQIACF_CF_STRUC_ENTRIES_MAX	1142	X'00000476'
MQIACF_CF_STRUC_ENTRIES_USED	1143	X'00000477'
MQIACF_CF_STRUC_BACKUP_SIZE	1144	X'00000478'
MQIACF_MOVE_TYPE	1145	X'00000479'
MQIACF_MOVE_TYPE_MOVE	1146	X'0000047A'
MQIACF_MOVE_TYPE_ADD	1147	X'0000047B'
MQIACF_Q_MGR_NUMBER	1148	X'0000047C'
MQIACF_Q_MGR_STATUS	1149	X'0000047D'

Tableau 185. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQIACF_DB2_CONN_STATUS	1150	X'0000047E'
MQIACF_SECURITY_ATTRS	1151	X'0000047F'
DÉLAI_SÉCURITÉ_MQIACF_MQI	1152	X'00000480'
INTERVALLE_SÉCURITÉ_MQIACF_MQI	1153	X'00000481'
MQIACF_SECURITY_SWITCH	1154	X'00000482'
PARAMÈTRE_SÉCURITÉ_MQI	1155	X'00000483'
MQIACF_STORAGE_CLASS_ATTRS	1156	X'00000484'
TYPE_MQIACF_USAGE_	1157	X'00000485'
ID_POOL_BUFFER_MQIACF_	1158	X'00000486'
MQIACF_USAGE_TOTAL_PAGES	1159	X'00000487'
PAGES_UTILISATION_MQIACF_UNUSAGE_UNUSED_PAGES	1160	X'00000488'
MQIACF_USAGE_PERSIST_PAGES	1161	X'00000489'
MQIACF_USAGE_NONPERSIST_PAGES	1162	X'0000048A'
MQIACF_USAGE_RESTART_EXTENTS	1163	X'0000048B'
MQIACF_NOMBRE_EXTENSIF_UTILISATION	1164	X'0000048C'
STATUT DE LA PAGE_MQIACF_PAGESET_STATUS	1165	X'0000048D'
MQIACF_USAGE_TOTAL_BUFFERS	1166	X'0000048E'
MQIACF_USAGE_DATA_SET_TYPE	1167	X'0000048F'
MQIACF_USAGE_PAGESET	1168	X'00000490'
MQIACF_USAGE_DATA_SET	1169	X'00000491'
MQIACF_POOL_BUFFER_USAGE_MQIACF_	1170	X'00000492'
MQIACF_MOVE_COUNT	1171	X'00000493'
MQIACF_EXPIRY_Q_COUNT	1172	X'00000494'
Objets de configuration_MQIACF_CONFIGURATION_OBJECTS	1173	X'00000495'
MQIACF_CONFIGURATION_EVENTS	1174	X'00000496'
TYPE_SYSP_MQIACF	1175	X'00000497'
MQIACF_SYSP_DEALLOC_INTERVAL	1176	X'00000498'
MQIACF_SYSP_MAX_ARCHIVE	1177	X'00000499'
MQIACF_SYSP_MAX_READ_TAPES	1178	X'0000049A'
MQIACF_SYSP_IN_BUFFER_SIZE	1179	X'0000049B'
MQIACF_SYSP_OUT_BUFFER_SIZE	1180	X'0000049C'
MQIACF_SYSP_OUT_BUFFER_COUNT	1181	X'0000049D'
MQIACF_SYSP_ARCHIVE	1182	X'0000049E'
MQIACF_SYSP_DUAL_ACTIVE	1183	X'0000049F'
MQIACF_SYSP_DUAL_ARCHIVE	1184	X'000004A0'
MQIACF_SYSP_DUAL_BSDS	1185	X'000004A1'
MQIACF_SYSP_MAX_CONNS	1186	X'000004A2'
MQIACF_SYSP_MAX_CONNS_AVANT	1187	X'000004A3'
MQIACF_SYSP_MAX_CONNS_BACK	1188	X'000004A4'

Tableau 185. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
INTERVALLE_EXIT_SYSP_MQIACF_XX_ENCODE_CASE_CAPS_LOCK_OFF	1189	X'000004A5'
MQIACF_SYSP_EXIT_TASKS	1190	X'000004A6'
MQIACF_SYSP_CHKPOINT_COUNT	1191	X'000004A7'
INTERVALLE_OTMA_SYSP_MQIACF_	1192	X'000004A8'
MQIACF_SYSP_Q_INDEX_DIFFÉRER	1193	X'000004A9'
MQIACF_SYSP_DB2_TASKS	1194	X'000004AA'
MQIACF_SYSP_RESLEVEL_AUDIT	1195	X'000004AB'
CODE MQIACF_SYSP_ROUTING_CODE	1196	X'000004AC'
MQIACF_SYSP_SMF_COMPTABILITÉ	1197	X'000004AD'
MQIACF_SYSP_SMF_STATS	1198	X'000004AE'
MQIACF_SYSP_SMF_INTERVAL	1199	X'000004AF'
MQIACF_SYSP_TRACE_CLASS	1200	X'000004B0'
TAILLE DU FICHER MQIACF_SYSP_TRACE_SIZE	1201	X'000004B1'
MQIACF_SYSP_INTERVALLE_WLM	1202	X'000004B2'
MQIACF_SYSP_ALLOC_UNIT	1203	X'000004B3'
MQIACF_SYSP_ARCHIVE_RETAIN	1204	X'000004B4'
MQIACF_SYSP_ARCHIVE_WTOR	1205	X'000004B5'
TAILLE_BLOC_SYSP_MQIACF_XX_ENCODE_CASE_CAPS_LOCK_OFF	1206	X'000004B6'
MQIACF_SYSP_CATALOGUE	1207	X'000004B7'
MQIACF_SYSP_COMPACT	1208	X'000004B8'
MQIACF_SYSP_ALLOC_PRIMARY	1209	X'000004B9'
MQIACF_SYSP_ALLOC_SECONDARY	1210	X'000004BA'
MQIACF_SYSP_PROTECT	1211	X'000004BB'
MQIACF_SYSP QUIESCE_INTERVAL	1212	X'000004BC'
HORODATAGE_SYSP_MQIACF_XX_ENCODE_CASE_CAPS_LOCK_OFF	1213	X'000004BD'
ADRESSE_UNIT_SYSP_MQIACF	1214	X'000004BE'
STATUT DU SYSTEME MQIACF_SYSP_UNIT_STATUS	1215	X'000004BF'
MQIACF_SYSP_LOG_COPY	1216	X'000004C0'
MQIACF_SYSP_LOG_UTILISÉ	1217	X'000004C1'
MQIACF_SYSP_LOG_SUSPEND	1218	X'000004C2'
MQIACF_SYSP_OFFLOAD_STATUS	1219	X'000004C3'
MQIACF_SYSP_TOTAL_LOGS	1220	X'000004C4'
MQIACF_SYSP_FULL_LOGS	1221	X'000004C5'
MQIACF_LISTENER_ATTRS	1222	X'000004C6'
MQIACF_LISTENER_STATUS_ATTRS	1223	X'000004C7'
MQIACF_SERVICE_ATTRS	1224	X'000004C8'
MQIACF_SERVICE_STATUS_ATTRS	1225	X'000004C9'
INDICATEUR_HEURE_Q_MQIACF_FILE	1226	X'000004CA'

Tableau 185. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQIACF_OLDEST_MSG_AGE	1227	X'000004CB'
OPTIONS D'AUTEUR MQIACF	1228	X'000004CC'
MQIACF_Q_MGR_STATUS_ATTRS	1229	X'000004CD'
MQIACF_CONNECTION_COUNT	1230	X'000004CE'
MQIACF_Q_MGR_FACILITY	1231	X'000004CF'
MQIACF_CHINIT_STATUS	1232	X'000004D0'
MQIACF_CMD_SERVER_STATUS	1233	X'000004D1'
DÉTAILLEMENT_ROUTE_MQIACF_ACF	1234	X'000004D2'
ACTIVITÉS MQIACF_RECORDED_ACTIVITÉS	1235	X'000004D3'
ACTIVITÉS MQIACF_MAX_ACTIVITÉS	1236	X'000004D4'
MQIACF_NOMBRE_DISCONTINUITÉS	1237	X'000004D5'
ACCUMULATION_ROUTE_MQIACF_XX_ENCODE_CASE_CAPS_LOCK_OFF	1238	X'000004D6'
MQIACF_ROUTE_DISTRIBUTION	1239	X'000004D7'
TYPE_OPÉRATION_MQIACF_OPÉRATION	1240	X'000004D8'
MQIACF_BACKOUT_COUNT	1241	X'000004D9'
CODE COMP_MQIACF	1242	X'000004DA'
CODAGE_MQIACF_CODAGE	1243	X'000004DB'
EXPIRATION_MQIACF_EXPIRATION	1244	X'000004DC'
MQIACF_FEEDBACK	1245	X'000004DD'
MQIACF_MSG_FLAGS	1247	X'000004DF'
LONGUE_MSG_MQIACF_MESSAGE	1248	X'000004E0'
TYPE MQIACF_MSG_	1249	X'000004E1'
DÉCALAGE_MQIACF_ACF	1250	X'000004E2'
MQIACF_ORIGINAL_LENGTH	1251	X'000004E3'
PERSISTENCE MQIACF_PERSISTANCE	1252	X'000004E4'
PRIO_MQIACF_ACF	1253	X'000004E5'
MQIACF_CODE_RAISON	1254	X'000004E6'
RAPPORT MQIACF_RAPPORT	1255	X'000004E7'
VERSION MQIACF_MQ	1256	X'000004E8'
ACTIVITÉS MQIACF_UNRECORDED_ACTIVITÉS	1257	X'000004E9'
MQIACF_SURVEILLANCE	1258	X'000004EA'
RÉACHEMINEMENT_ROUTE_MQIACF_ACF	1259	X'000004EB'
STATUT DU SERVICE MQIACF_STATUT	1260	X'000004EC'
MQIACF_Q_TYPES	1261	X'000004ED'
MQIACF_XX_ENCODE_CASE_ONE id_utilisateur_support	1262	X'000004EE'
VERSION INTERFACF_MQIACF	1263	X'000004EF'
MQIACF_AUTH_SERVICE_ATTRS	1264	X'000004F0'
TYPE_EXTENSIF_MQIACF_USAGE_	1265	X'000004F1'
MQIACF_SYSP_CLUSTER_CACHE	1266	X'000004F2'


Tableau 185. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQIACF_SYSP_DB2_BLOB_TASKS	1267	X'000004F3'
MQIACF_SYSP_WLM_INT_UNITS	1268	X'000004F4'
MQIACF_TOPIC_ATTRS	1269	X'000004F5'
MQIACF_PUBSUB_PROPERTIES	1271	X'000004F7'
CLASSE_DESTINATION_MQIACF_ACF	1273	X'000004F9'
MQIACF_DURABLE_ABONNEMENT	1274	X'000004FA'
MQIACF_SUBSCRIPTION_SCOPE	1275	X'000004FB'
ID_UTILISATEUR_MQIACF_VARIABLE_	1277	X'000004FD'
MQIACF_REQUEST_ONLY	1280	X'00000500'
MQIACF_PRIORITE de publication	1283	X'00000503'
MQIACF_SUB_ATTRS	1287	X'00000507'
SCHÉMA_GÉNÉRIQUES-MQIACF_WILDCARD_SCHEMA	1288	X'00000508'
SOUS-TYPE_MQIACF_	1289	X'00000509'
NOMBRE_MESSAGE_MQIACF_MESSAGE	1290	X'0000050A'
MQIACF_Q_MGR_PUBSUB	1291	X'0000050B'
MQIACF_Q_MGR_VERSION	1292	X'0000050C'
MQIACF_SUB_STATUS_ATTRS	1294	X'0000050E'
STATUT DE LA COMMANDE_MQIACF_TOPIC_STATUS	1295	X'0000050F'
MQIACF_TOPIC_SUB	1296	X'00000510'
MQIACF_TOPIC_PUB	1297	X'00000511'
MQIACF_RETAINED_PUBLICATION	1300	X'00000514'
MQIACF_TOPIC_STATUS_ATTRS	1301	X'00000515'
MQIACF_TOPIC_STATUS_TYPE	1302	X'00000516'
MQIACF_SUB_OPTIONS	1303	X'00000517'
NOMBRE_DIFF_MQIACF_PUBLICATION	1304	X'00000518'
MQIACF_CLEAR_TYPE	1305	X'00000519'
Portée_MQIACF_CLEAR_SCOPE	1306	X'0000051A'
MQIACF_SOUS-NIVEAU	1307	X'0000051B'
MQIACF_ASYNC_STATE	1308	X'0000051C'
MQIACF_SYNTHESE	1309	X'0000051D'
MSGS_OBSOLÈTES_MQIACF_	1310	X'0000051E'
MQIACF_PUBSUB_STATUS	1311	X'0000051F'
MQIACF_PS_STATUS_TYPE	1314	X'00000522'
MQIACF_PUBSUB_STATUS_ATTRS	1318	X'00000526'
MQIACF_SELECTOR_TYPE	1321	X'00000529'
MQIACF_MCAST_REL_INDICATEUR	1351	X'00000547'
MQIACF_CHLAUTH_TYPE	1352	X'00000548'
MQXR_TYPE_DIAGNOSTIC	1354	X'0000054A'
MQIACF_CHLAUTH_ATTRS	1355	X'0000054B'

Tableau 185. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
ID_OPERATION_MQIACF	1356	X'0000054C'
MQIACF_API_CALLER_TYPE	1357	X'0000054D'
ENVIRONNEMENT MQIACF_API	1358	X'0000054E'
MQIACF_TRACE_DETAIL	1359	X'0000054F'
MQIACF_HOBJ	1360	X'00000550'
MQIACF_TYPE_APPEL	1361	X'00000551'
Opération MQIACF_MQCB_OPERATION	1362	X'00000552'
TYPE MQIACF_MQCB_	1363	X'00000553'
OPTIONS MQIACF_MQCB_	1364	X'00000554'
OPTIONS MQIACF_CLOSE_OPTIONS	1365	X'00000555'
OPERATION MQIACF_CTL_MQIACF	1366	X'00000556'
OPTIONS MQIACF_GET_OPTIONS	1367	X'00000557'
MQIACF_RECS_PRESENT	1368	X'00000558'
MQIACF_KNOWN_DEST_COUNT	1369	X'00000559'
MQIACF_UNKNOWN_DEST_COUNT	1370	X'0000055A'
MQIACF_INVALID_DEST_COUNT	1371	X'0000055B'
MQIACF_RESOLVED_TYPE	1372	X'0000055C'
OPTIONS MQIACF_PUT_MQ	1373	X'0000055D'
LONGUEUR_BUFFER_MQIACF	1374	X'0000055E'
LONGUEUR_DONNÉES_TRACE_MQIACF_	1375	X'0000055F'
MQIACF_SMDS_EXPANDST	1376	X'00000560'
LONGUEUR_STRUC_MQIACF_ACF	1377	X'00000561'
NOMBRE_ÉLÉMENTS_MQI	1378	X'00000562'
MQIACF_EXPIRATION_HEURE	1379	X'00000563'
MQIACF_HEURE_CONNEXION	1380	X'00000564'
MQIACF_DISCONNECT_TIME	1381	X'00000565'
MQIACF_HSUB	1382	X'00000566'
OPTIONS MQIACF_SUBRQ_MQ	1383	X'00000567'
MQIACF_XA_RMID	1384	X'00000568'
MQIACF_XA_FLAGS	1385	X'00000569'
MQIACF_XA_RETCODE	1386	X'0000056A'
MQIACF_XA_HANDLE	1387	X'0000056B'
MQIACF_XA_RETVAL	1388	X'0000056C'
TYPE_STATUS_MQIACF	1389	X'0000056D'
MQIACF_XA_COUNT	1390	X'0000056E'
MQIACF_SELECTOR_COUNT	1391	X'0000056F'
Sélecteurs MQIACF_SELECTORS	1392	X'00000570'
MQIACF_INTATTR_COUNT	1393	X'00000571'
MQIACF_INTATTRS	1394	X'00000572'

Tableau 185. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQIACF_SUBRQ_ACTION	1395	X'00000573'
MQIACF_NUM_PUBS	1396	X'00000574'
TAILLE MQIACF_POINTER_taille	1397	X'00000575'
Instruction MQIACF_REMOVE_AUTHREC	1398	X'00000576'
MQIACF_XR_ATTRS	1399	X'00000577'
TYPE_FONCTION_APPL_MQIACF_XX_ENCODE_CASE_CAPS_LOCK_OFF	1400	X'00000578'
MQIACF_AMQP_ATTRS	1401	X'00000579'
MQIACF_EXPORT_TYPE	1402	X'0000057A'
MQIACF_EXPORT_ATTRS	1403	X'0000057B'
OBJETS_SYSTÈME_MQIACF_XX_ENCODE_CASE_CAPS_LOCK_OFF	1404	X'0000057C'
MQIACF_CONNECTION_SWAP	1405	X'0000057D'
MQIACF_AMQP_DIAGNOSTICS_TYPE	1406	X'0000057E'
EMPLACEMENT_POOL_BUFFER_MQIACF_	1408	X'00000580'
STATUT_CONNEXION_LDAP_MQIACF_	1409	X'00000581'
MQIACF_SYSP_MAX_ACE_POOL	1410	X'00000582'
MQIACF_PAGECLAS	1411	X'00000583'
MQIACF_AUTH_REC_TYPE	1412	X'00000584'
MQIACF_SYSP_MAX_CONC_OFFLOADS	1413	X'00000585'
MQIACF_SYSP_ZHYPERWRITE	1414	X'00000586'
MQIACF_Q_MGR_STATUS_LOG	1415	X'00000587'
TAILLE_JOURNAL_ARCHIVE_MQIACF_XX_ENCODE_CASE_CAPS_LOCK_OFF	1416	X'00000588'
TAILLE DE JOURNAL MOYENNE MQIACF_	1417	X'00000589'
TAILLE_JOURNAL_REDÉMARRE_MQIACF_	1418	X'0000058A'
MQIACF_REUSABLE_LOG_SIZE	1419	X'0000058B'
MQIACF_LOG_IN_USE	1420	X'0000058C'
UTILISATION_JOURNAL_MQIACF_XX_ENCODE_CASE_CAPS_LOCK_OFF	1421	X'0000058D'
MQIACF_IGNORE_STATE	1423	X'0000058F'
MQIACF_MOVABLE_APPL_COUNT	1424	X'00000590'
MQIACF_APPL_INFO_ATTRS	1425	X'00000591'
MQIACF_APPL_MOBILE	1426	X'00000592'
MQIACF_REMOTE_QMGR_ACTIVE	1427	X'00000593'
MQIACF_APPL_INFO_TYPE	1428	X'00000594'
MQIACF_APPL_INFO_APPL	1429	X'00000595'
MQIACF_APPL_INFO_QMGR	1430	X'00000596'
MQIACF_APPL_INFO_LOCAL	1431	X'00000597'
MQIACF_APPL_IMMOVABLE_COUNT	1432	X'00000598'
MQIACF_EQUILIBRÉ	1433	X'00000599'
MQIACF_ETAT_BALSTATE	1434	X'0000059A'



Tableau 185. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQIACF_APPL_IMMOVABLE_REASON	1435	X'0000059B'
MQIACF_DS_ENCRYPTED	1436	X'0000059C'
TAILLE DE FILE D'ATTENTE MQIACF_CUR_Q_FILE_SIZE	1437	X'0000059D'
TAILLE DU FICHIER MQIACF_CUR_MAX_FILE_SIZE	1438	X'0000059E'
MQIACF_BALANCING_TYPE	1439	X'0000059F'
MQIACF_BALANCING_OPTIONS	1440	X'000005A0'
DÉLAI_EXPIRATION_BALANCE_MQIACF	1441	X'000005A1'
MQIACF_SYSP_SMF_STAT_TIME_SECS	1442	X'000005A2'
MQIACF_SYSP_SMF_ACCT_TIME_MINS	1443	X'000005A3'
MQIACF_SYSP_SMF_ACCT_TIME_SECS	1444	X'000005A4'
 MQIACF_LAST_USED	1444	X'000005A4'

### MQIACH\_\* (types de canal de type entier au format de commande)

Tableau 186. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQIACH_FIRST	1501	X'000005DD'
MQIACH_XMIT_PROTOCOL_TYPE	1501	X'000005DD'
MQIACH_BATCH_SIZE	1502	X'000005DE'
MQIACH_DISC_INTERVAL	1503	X'000005DF'
MQIACH_SHORT_TIMER	1504	X'000005E0'
MQIACH_SHORT_RETRY	1505	X'000005E1'
MQIACH_LONG_TIMER	1506	X'000005E2'
MQIACH_LONG_RETRY	1507	X'000005E3'
MQIACH_PUT_AUTHORITY	1508	X'000005E4'
MQIACH_SEQUENCE_NUMBER_WRAP	1509	X'000005E5'
MQIACH_MAX_MSG_LENGTH	1510	X'000005E6'
MQIACH_CHANNEL_TYPE	1511	X'000005E7'
MQIACH_DATA_COUNT	1512	X'000005E8'
MQIACH_NAME_COUNT	1513	X'000005E9'
MQIACH_MSG_SEQUENCE_NUMBER	1514	X'000005EA'
MQIACH_DATA_CONVERSION	1515	X'000005EB'
MQIACH_IN_DOUBT	1516	X'000005EC'
MQIACH_MCA_TYPE	1517	X'000005ED'
MQIACH_SESSION_COUNT	1518	X'000005EE'
MQIACH_ADAPTER	1519	X'000005EF'
MQIACH_COMMAND_COUNT	1520	X'000005F0'
MQIACH_SOCKET	1521	X'000005F1'
MQIACH_PORT	1522	X'000005F2'
MQIACH_CHANNEL_INSTANCE_TYPE	1523	X'000005F3'

Tableau 186. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQIACH_CHANNEL_INSTANCE_ATTRS	1524	X'000005F4'
MQIACH_CHANNEL_ERROR_DATA	1525	X'000005F5'
MQIACH_CHANNEL_TABLE	1526	X'000005F6'
MQIACH_CHANNEL_STATUS	1527	X'000005F7'
MQIACH_INDOUBT_STATUS	1528	X'000005F8'
MQIACH_LAST_SEQ_NUMBER	1529	X'000005F9'
MQIACH_LAST_SEQUENCE_NUMBER	1529	X'000005F9'
MQIACH_CURRENT_MSGS	1531	X'000005FB'
MQIACH_CURRENT_SEQ_NUMBER	1532	X'000005FC'
MQIACH_CURRENT_SEQUENCE_NUMBER	1532	X'000005FC'
MQIACH_SSL_RETURN_CODE	1533	X'000005FD'
MQIACH_MSGS	1534	X'000005FE'
MQIACH_BYTES_SENT	1535	X'000005FF'
MQIACH_BYTES_RCVD	1536	X'00000600'
MQIACH_BYTES_RECEIVED	1536	X'00000600'
MQIACH_BATCHES	1537	X'00000601'
MQIACH_BUFFERS_SENT	1538	X'00000602'
MQIACH_BUFFERS_RCVD	1539	X'00000603'
MQIACH_BUFFERS_RECEIVED	1539	X'00000603'
MQIACH_LONG_RETRIES_LEFT	1540	X'00000604'
MQIACH_SHORT_RETRIES_LEFT	1541	X'00000605'
MQIACH_MCA_STATUS	1542	X'00000606'
MQIACH_STOP_REQUESTED	1543	X'00000607'
MQIACH_MR_COUNT	1544	X'00000608'
MQIACH_MR_INTERVAL	1545	X'00000609'
MQIACH_NPM_SPEED	1562	X'0000061A'
MQIACH_HB_INTERVAL	1563	X'0000061B'
MQIACH_BATCH_INTERVAL	1564	X'0000061C'
MQIACH_NETWORK_PRIORITY	1565	X'0000061D'
MQIACH_KEEP_ALIVE_INTERVAL	1566	X'0000061E'
MQIACH_BATCH_HB	1567	X'0000061F'
MQIACH_SSL_CLIENT_AUTH	1568	X'00000620'
MQIACH_ALLOC_RETRY	1570	X'00000622'
MQIACH_ALLOC_FAST_TIMER	1571	X'00000623'
MQIACH_ALLOC_SLOW_TIMER	1572	X'00000624'
MQIACH_DISC_RETRY	1573	X'00000625'
MQIACH_PORT_NUMBER	1574	X'00000626'
MQIACH_HDR_COMPRESSION	1575	X'00000627'
MQIACH_MSG_COMPRESSION	1576	X'00000628'

Tableau 186. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQIACH_CLWL_CHANNEL_RANK	1577	X'00000629'
MQIACH_CLWL_CHANNEL_PRIORITY	1578	X'0000062A'
MQIACH_CLWL_CHANNEL_WEIGHT	1579	X'0000062B'
MQIACH_CHANNEL_DISP	1580	X'0000062C'
MQIACH_INBOUND_DISP	1581	X'0000062D'
MQIACH_CHANNEL_TYPES	1582	X'0000062E'
MQIACH_ADAPS_STARTED	1583	X'0000062F'
MQIACH_ADAPS_MAX	1584	X'00000630'
MQIACH_DISPS_STARTED	1585	X'00000631'
MQIACH_DISPS_MAX	1586	X'00000632'
MQIACH_SSLTASKS_STARTED	1587	X'00000633'
MQIACH_SSLTASKS_MAX	1588	X'00000634'
MQIACH_CURRENT_CHL	1589	X'00000635'
MQIACH_CURRENT_CHL_MAX	1590	X'00000636'
MQIACH_CURRENT_CHL_TCP	1591	X'00000637'
MQIACH_CURRENT_CHL_LU62	1592	X'00000638'
MQIACH_ACTIVE_CHL	1593	X'00000639'
MQIACH_ACTIVE_CHL_MAX	1594	X'0000063A'
MQIACH_ACTIVE_CHL_PAUSED	1595	X'0000063B'
MQIACH_ACTIVE_CHL_STARTED	1596	X'0000063C'
MQIACH_ACTIVE_CHL_STOPPED	1597	X'0000063D'
MQIACH_ACTIVE_CHL_RETRY	1598	X'0000063E'
MQIACH_LISTENER_STATUS	1599	X'0000063F'
MQIACH_SHARED_CHL_RESTART	1600	X'00000640'
MQIACH_LISTENER_CONTROL	1601	X'00000641'
MQIACH_BACKLOG	1602	X'00000642'
MQIACH_XMITQ_TIME_INDICATOR	1604	X'00000644'
MQIACH_NETWORK_TIME_INDICATOR	1605	X'00000645'
MQIACH_EXIT_TIME_INDICATOR	1606	X'00000646'
MQIACH_BATCH_SIZE_INDICATOR	1607	X'00000647'
MQIACH_XMITQ_MSGS_AVAILABLE	1608	X'00000648'
MQIACH_CHANNEL_SUBSTATE	1609	X'00000649'
MQIACH_SSL_KEY_RESETS	1610	X'0000064A'
MQIACH_COMPRESSION_RATE	1611	X'0000064B'
MQIACH_COMPRESSION_TIME	1612	X'0000064C'
MQIACH_MAX_XMIT_SIZE	1613	X'0000064D'
MQIACH_DEF_CHANNEL_DISP	1614	X'0000064E'
MQIACH_SHARING_CONVERSATIONS	1615	X'0000064F'
MQIACH_MAX_SHARING_CONVS	1616	X'00000650'

Tableau 186. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQIACH_CURRENT_SHARING_CONVS	1617	X'00000651'
MQIACH_MAX_INSTANCES	1618	X'00000652'
MQIACH_MAX_INSTS_PER_CLIENT	1619	X'00000653'
MQIACH_CLIENT_CHANNEL_WEIGHT	1620	X'00000654'
MQIACH_CONNECTION_AFFINITY	1621	X'00000655'
MQIACH_AUTH_INFO_TYPES	1622	X'00000656'
MQIACH_RESET_REQUESTED	1623	X'00000657'
MQIACH_BATCH_DATA_LIMIT	1624	X'00000658'
MQIACH_MSG_HISTORY	1625	X'00000659'
MQIACH_MULTICAST_PROPERTIES	1626	X'0000065A'
MQIACH_NEW_SUBSCRIBER_HISTORY	1627	X'0000065B'
MQIACH_MC_HB_INTERVAL	1628	X'0000065C'
MQIACH_USE_CLIENT_ID	1629	X'0000065D'
MQIACH_MQTT_KEEP_ALIVE	1630	X'0000065E'
MQIACH_IN_DOUBT_IN	1631	X'0000065F'
MQIACH_IN_DOUBT_OUT	1632	X'00000660'
MQIACH_MSGS_SENT<	1633	X'00000661'
MQIACH_MSGS_RECEIVED	1634	X'00000662'
MQIACH_MSGS_RCVD	1634	X'00000662'
MQIACH_PENDING_OUT	1635	X'00000663'
MQIACH_AVAILABLE_CIPHERSPECS	1636	X'00000664'
MQIACH_MATCH	1637	X'00000665'
MQIACH_USER_SOURCE	1638	X'00000666'
MQIACH_WARNING	1639	X'00000667'
MQIACH_DEF_RECONNECT	1640	X'00000668'
MQIACH_CHANNEL_SUMMARY_ATTRS	1642	X'0000066A'
MQIACH_PROTOCOL	1643	X'0000066B'
MQIACH_AMQPKEEPALIVE	1644	X'0000066C'
MQIACH_SECURITY_PROTOCOL	1645	X'0000066D'
 MQIACH_SPL_PROTECTION	1646	X'0000066E'
MQIACH_LAST_USED	1646	X'0000066E'

## MQIAMO\_\* (format de commande-Types de paramètre de surveillance des entiers)

Tableau 187. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQIAMO_FIRST	701	X'000002BD'
TAILLE DE LA COMMANDE MQIAMO_AVG_BATCH_SIZE	702	X'000002BE'
MQIAMO_AVG_Q_HEURE	703	X'000002BF'

Tableau 187. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQIAMO_ANNULATIONS	704	X'000002C0'
MQIAMO_NAVIGATIONS	705	X'000002C1'
MQIAMO_BROWSE_MAX_BYTES	706	X'000002C2'
MQIAMO_BROWSE_MIN_BYTES	707	X'000002C3'
Echec de MQIAMO_BROWSES_FAILED	708	X'000002C4'
MQIAMO_FERMETURES	709	X'000002C5'
MQIAMO_VALIDATIONS	710	X'000002C6'
Echec de MQIAMO_COMMITS_FAILED	711	X'000002C7'
MQIAMO_CONNS	712	X'000002C8'
MQIAMO_CONNS_MAX	713	X'000002C9'
MQIAMO_DISQUES	714	X'000002CA'
MQIAMO_DISCS_IMPLICITE	715	X'000002CB'
TYPE_DISC_MQIAMO	716	X'000002CC'
MQIAMO_EXIT_TIME_AVG	717	X'000002CD'
MQIAMO_EXIT_TIME_MAX	718	X'000002CE'
MQIAMO_EXIT_TEMPS_MIN	719	X'000002CF'
MQIAMO_LOTS	720	X'000002D0'
MQIAMO_MSGS	721	X'000002D1'
MQIAMO_GETS	722	X'000002D2'
MQIAMO_GET_MAX_BYTES	723	X'000002D3'
MQIAMO_GET_MIN_BYTES	724	X'000002D4'
Echec de MQIAMO_GETS_FAILED	725	X'000002D5'
MQIAMO_INCOMPLETE_LOTS	726	X'000002D6'
MQIAMO_INQS	727	X'000002D7'
MQIAMO_MSGS	728	X'000002D8'
MQIAMO_NET_TIME_AVG	729	X'000002D9'
MQIAMO_TEMPS_NET_MAX	730	X'000002DA'
MQIAMO_HEURE_NET_MIN	731	X'000002DB'
NOMBRE_OBJETS_MQIAMO_OBJET	732	X'000002DC'
MQIAMO_OUVERTURES	733	X'000002DD'
MQIAMO_PUT1S	734	X'000002DE'
MQIAMO_INSERTIONS	735	X'000002DF'
MQIAMO_PUT_MAX_BYTES	736	X'000002E0'
MQIAMO_PUT_MIN_BYTES	737	X'000002E1'
MQIAMO_PUT_RETRIES	738	X'000002E2'
MQIAMO_Q_MAX_PROFONDEUR	739	X'000002E3'
MQIAMO_Q_MIN_DEPTH	740	X'000002E4'
MQIAMO_Q_TIME_MOY	741	X'000002E5'
MQIAMO_Q_TIME_MAX	742	X'000002E6'

Tableau 187. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQIAMO_HEURE_Q_MIN	743	X'000002E7'
MQIAMO_SETS	744	X'000002E8'
Echec de MQIAMO_CONNS_FAILED	749	X'000002ED'
Echec de MQIAMO_OPENS_FAILED	751	X'000002EF'
Echec de MQIAMO_INQS_FAILED	752	X'000002F0'
MQIAMO_SETS_FAILED	753	X'000002F1'
MQIAMO_PUTS_FAILED	754	X'000002F2'
MQIAMO_PUT1S_FAILED	755	X'000002F3'
Echec de MQIAMO_CLOSES_FAILED	757	X'000002F5'
MQIAMO_MSGS_EXPIRED	758	X'000002F6'
MQIAMO_MSGS_NOT_QUEUED	759	X'000002F7'
MQIAMO_MSGS_PURGED	760	X'000002F8'
MQIAMO_SUBS_DUR	764	X'000002FC'
MQIAMO_SUBS_NDUR	765	X'000002FD'
Echec de MQIAMO_SUBS_FAILED	766	X'000002FE'
MQIAMO_SUBRQS	767	X'000002FF'
Echec de MQIAMO_SUBRQS_FAILED	768	X'00000300'
MQIAMO_CBS	769	X'00000301'
MQIAMO_CBS_ECHEC	770	X'00000302'
MQIAMO_CTLs	771	X'00000303'
MQIAMO_CTLs_FAILED	772	X'00000304'
MQIAMO_STATS	773	X'00000305'
MQIAMO_STATS_ECHEC	774	X'00000306'
MQIAMO_SUB_DUR_HIGHWATER	775	X'00000307'
MQIAMO_SUB_DUR_LOWWATER	776	X'00000308'
MQIAMO_SUB_NDUR_HIGHWATER	777	X'00000309'
MQIAMO_SUB_NDUR_LOWWATER	778	X'0000030A'
MQIAMO_TOPIC_PUTS	779	X'0000030B'
MQIAMO_TOPIC_PUTS_FAILED	780	X'0000030C'
MQIAMO_TOPIC_PUT1S	781	X'0000030D'
MQIAMO_TOPIC_PUT1S_FAILED	782	X'0000030E'
MQIAMO_NOMBRE_MSG_PUBLICATION	784	X'00000310'
MQIAMO_UNSUBS_DUR	786	X'00000312'
MQIAMO_UNSUBS_NDUR	787	X'00000313'
MQIAMO_UNSUBS_FAILED	788	X'00000314'
INTERVALLE_MQIAMO_	789	X'00000315'
MQIAMO_MSGS_SENT	790	X'00000316'
MQIAMO_BYTES_SENT	791	X'00000317'
MQIAMO_REPAIR_BYTES	792	X'00000318'

Tableau 187. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MODE_PASSE_ALIMENTATION_MQIAMO_MODE	793	X'00000319'
MQIAMO_TYPE_FIABILITÉ	794	X'0000031A'
MQIAMO_LATE_JOIN_MARK	795	X'0000031B'
MQIAMO_NACKS_RCVD	796	X'0000031C'
MQIAMO_REPAIR_PKTS	797	X'0000031D'
MQIAMO_HISTORY_PKTS	798	X'0000031E'
MQIAMO_PENDING_PKTS	799	X'0000031F'
MQIAMO_PKT_RATE	800	X'00000320'
MQIAMO_MCAST_XMIT_RATE	801	X'00000321'
MQIAMO_MCAST_BATCH_HEURE	802	X'00000322'
MQIAMO_MCAST_HEARTBEAT	803	X'00000323'
PORT MQIAMO_DEST_DATA_PORT	804	X'00000324'
PORT MQIAMO_DEST_REPAIR_PORT	805	X'00000325'
MQIAMO_ACKS_RCVD	806	X'00000326'
MQIAMO_ACTIVE_ACKERS	807	X'00000327'
MQIAMO_PKTS_SENT	808	X'00000328'
MQIAMO_TOTAL_REPAIR_PKTS	809	X'00000329'
MQIAMO_TOTAL_PKTS_SENT	810	X'0000032A'
MQIAMO_TOTAL_MSGS_SENT	811	X'0000032B'
MQIAMO_TOTAL_BYTES_SENT	812	X'0000032C'
MQIAMO_NUM_STREAMS	813	X'0000032D'
MQIAMO_ACK_FEEDBACK	814	X'0000032E'
MQIAMO_NACK_FEEDBACK	815	X'0000032F'
MQIAMO_PKTS_LOST	816	X'00000330'
MQIAMO_MSGS_RCVD	817	X'00000331'
MQIAMO_MSG_BYTES_RCVD	818	X'00000332'
MQIAMO_MSGS_DELIVERED	819	X'00000333'
MQIAMO_PKTS_PROCESSED	820	X'00000334'
MQIAMO_PKTS_DLVD	821	X'00000335'
MQIAMO_PKTS_SUPPRIMÉ	822	X'00000336'
MQIAMO_PKTS_DUPLIQUÉ	823	X'00000337'
MQIAMO_NACKS_CREATED	824	X'00000338'
MQIAMO_NACK_PKTS_SENT	825	X'00000339'
MQIAMO_REPAIR_PKTS_RQSTD	826	X'0000033A'
MQIAMO_REPAIR_PKTS_RCVD	827	X'0000033B'
MQIAMO_PKTS_RÉPARÉ	828	X'0000033C'
MQIAMO_TOTAL_MSGS_RCVD	829	X'0000033D'
MQIAMO_TOTAL_MSGS_BYTES_RCVD	830	X'0000033E'
MQIAMO_TOTAL_REPAIR_PKTS_RCVD	831	X'0000033F'

Tableau 187. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQIAMO_TOTAL_REPAIR_PKTS_RQSTD	832	X'00000340'
MQIAMO_TOTAL_MSGS_PROCESSED	833	X'00000341'
MQIAMO_TOTAL_MSGS_SELECTED	834	X'00000342'
MQIAMO_TOTAL_MSGS_EXPIRED	835	X'00000343'
MQIAMO_TOTAL_MSGS_DELIVERED	836	X'00000344'
MQIAMO_TOTAL_MSGS_RETURNED	837	X'00000345'
MQIAMO_LAST_USED	837	X'00000345'

### **MQIAMO64\_\* (format de commande-Types de paramètre de surveillance des entiers 64 bits)**

Tableau 188. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQIAMO64_AVG_Q_TIME	703	X'000002BF'
MQIAMO64_Q_TIME_AVG	741	X'000002E5'
MQIAMO64_Q_TIME_MAX	742	X'000002E6'
MQIAMO64_Q_TIME_MIN	743	X'000002E7'
MQIAMO64_BROWSE_BYTES	745	X'000002E9'
MQIAMO64_BYTES	746	X'000002EA'
MQIAMO64_GET_BYTES	747	X'000002EB'
MQIAMO64_PUT_BYTES	748	X'000002EC'
MQIAMO64_TOPIC_PUT_BYTES	783	X'0000030F'
MQIAMO64_PUBLISH_MSG_BYTES	785	X'00000311'

### **MQIASY\_\* (Sélecteurs système entiers)**

Tableau 189. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQIASY_FIRST	-1	X'FFFFFFFF'
MQIASY_CODED_CHAR_SET_ID	-1	X'FFFFFFFF'
TYPE_MQIAS	-2	X'FFFFFFFE'
MQIASY_COMMAND	-3	X'FFFFFFFD'
NUMERO MQIASY_MSG_SEQ_NO	-4	X'FFFFFFFC'
CONTROLE MQIASY_	-5	X'FFFFFFFB'
CODE COMP_MQIASY_	-6	X'FFFFFFFA'
MQIASY_MOTIF	-7	X'FFFFFFF9'
OPTIONS MQIASY_BAG_OPTIONS	-8	X'FFFFFFF8'
VERSION MQIASY_VERSION	-9	X'FFFFFFF7'
MQIASY_LAST_USED	-9	X'FFFFFFF7'
MQIASY_LAST	-2000	X'FFFFFF830'



## MQIAUT\_\* (Authentificateur de l'en-tête d'information IMS)

Tableau 190. Noms et valeurs de constante	
Nom	Valeur
MQIAUT_AUCUN	"rrrrrrrrrr"
MQIAUT_NON_ARRAY	'r','r'

**Remarque :** Le symbole r représente un caractère blanc unique.

## MQIAV\_\* (Valeurs d'attribut d'entier)

Tableau 191. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQIAV_NOT_APPLICABLE	-1	X'FFFFFFFF'
MQIAV_NON DEFINI	-2	X'FFFFFFFE'

## MQICM\_\* (modes de validation de l'en-tête d'information IMS)

Tableau 192. Noms et valeurs de constante	
Nom	Valeur
MQICM_COMMIT_THEN_SEND	'0'
MQICM_SEND_THEN_COMMIT	'1'

## MQIDO\_\* (Options d'attente de validation du format de commande)

Tableau 193. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
VALIDATION_MQIDO_VALIDATION	1	X'00000001'
MQIDO_BACKOUT	2	X'00000002'

## MQIEP\_\* (points d'entrée d'interface)

### Structure des paramètres de sécurité de connexion

Tableau 194. Structures de constantes	
Nom	Structure
ID_STRUC_MQIPE	"IEP r"
MQIEP_STRUC_ID_ARRAY	'I','E','P','r'

**Remarque :** Le symbole r représente un caractère blanc unique.

Tableau 195. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQIEP_VERSION_1	1	X'00000001'
MQDXP_CURRENT_VERSION	1	X'00000001'

## MQIGQ\_\* (mise en file d'attente intra-groupe)

Tableau 196. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQIGQ_DISABLED	0	X'00000000'
MQIGQ_ENABLED	1	X'00000001'

## MQIGQPA\_\* (droit d'insertion dans la file d'attente d'un groupe interne)

Tableau 197. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQIGQPA_PAR DEF AUT	1	X'00000001'
MQIGQPA_CONTEXT	2	X'00000002'
MQIGQPA_ONLY_IGQ	3	X'00000003'
MQIGQPA_ALTERNATE_OR_IGQ	4	X'00000004'

## MQIIH\_\* (indicateurs et structure d'en-tête d'information IMS)

### Structure d'en-tête d'informations IMS

Tableau 198. Structures de constantes	
Nom	Structure
ID MQIIH_STRUC_ID	"IIH↵"
MQIIH_STRUC_ID_ARRAY	'I','I','H','↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 199. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQIIH_VERSION_1	1	X'00000001'
MQIIH_VERSION_CURRENT_	1	X'00000001'
MQIIH_LENGTH_1	84	X'00000054'

### Indicateurs d'en-tête d'informations IMS

Tableau 200. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQIIH_AUCUN	0	X'00000000'
MQIIH_PASS_EXPIRATION	1	X'00000001'
MQIIH_EXPIRATION_ILLIMITÉE	0	X'00000000'
MQIIH_REPLY_FORMAT_NONE	8	X'00000008'
MQIIH_IGNORE_PURG	16	X'00000010'
MQIIH_CM0_REQUEST_RESPONSE	32	X'00000020'

## MQIMPO\_\* (interrogation des options et de la structure des propriétés de message)

### Consulter la structure des options de propriété de message

Nom	Structure
ID_STRUC_MQIMPO_	"IMPO"
MQIMPO_STRUC_ID_ARRAY	'I', 'M', 'P', 'O'

**Remarque :** Le symbole – représente un caractère blanc unique.

Nom	Valeur décimale	Valeur hexadécimale
MQIMPO_VERSION_1	1	X'00000001'
MQIMPO_CURRENT_VERSION	1	X'00000001'

### Interroger les options de propriété de message

Nom	Valeur décimale	Valeur hexadécimale
MQIMPO_TYPE_CONVERT	2	X'00000002'
LONGUEUR_REQUÊTE_MQIMPO_LONG	4	X'00000004'
MQIMPO_INQ_FIRST	0	X'00000000'
MQIMPO_INQ_NEXT	8	X'00000008'
MQIMPO_INQ_PROP_UNDER_CURSOR	16	X'00000010'
MQIMPO_CONVERT_VALUE	32	X'00000020'
MQIMPO_NONE	0	X'00000000'

## MQINBD\_\* (Dispositions entrantes du format de commande)

Nom	Valeur décimale	Valeur hexadécimale
MQINBD_Q_DIR	0	X'00000000'
MQINBD_GROUPE	3	X'00000003'

## MQIND\_\* (Valeurs d'index spéciales)

Nom	Valeur décimale	Valeur hexadécimale
MQIND_AUCUN	-1	X'FFFFFFFF'
MQIND_ALL	-2	X'FFFFFFFE'

## MQIPADDR\_\* (versions d'adresse IP)

Nom	Valeur décimale	Valeur hexadécimale
MQIPADDR_IPV4	0	X'00000000'

Tableau 206. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQIPADDR_IPV6	1	X'00000001'

### MQISS\_\* (Portées de sécurité de l'en-tête d'information IMS)

Tableau 207. Noms et valeurs de constante	
Nom	Valeur
CONTROLE MQISS_CHECK	'C'
MQISS_FULL	'F'

### MQIT\_\* (types d'index)

Tableau 208. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQIT_AUCUN	0	X'00000000'
ID_MSG_MQ	1	X'00000001'
ID_CORREL_MQ	2	X'00000002'
JETON_MSG_MQIT_TOKEN	4	X'00000004'
ID_GROUPE_MQIT	5	X'00000005'

### MQITEM\_\* (Type d'élément pour mqInquireItemInfo)

Tableau 209. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQITEM_ENTIER	1	X'00000001'
MQITEM_CHAINE	2	X'00000002'
COUSSIN MQITEM_BAG	3	X'00000003'
MQITEM_BYTE_STRING	4	X'00000004'
FILTRE_ENTIER_MQITEM_ENTIER	5	X'00000005'
FILTRE_CHAÎNE_MQITEM_FILTER	6	X'00000006'
MQITEM_INTEGER64	7	X'00000007'
MQITEM_BYTE_STRING_FILTER	8	X'00000008'

### MQITII\_\* (identificateur d'instance de transaction d'en-tête d'informations IMS)

Tableau 210. Noms et valeurs de constante	
Nom	Valeur
MQITII_AUCUN	X'00...00' (16 valeurs nulles)
MQITII_NON_ARRAY	'\0', '\0', ... (16 valeurs nulles)

### MQITS\_\* (en-tête d'informations IMS -Etats de transaction)

Tableau 211. Noms et valeurs de constante	
Nom	Valeur
MQITS_DANS_CONVERSATION	'C'

Tableau 211. Noms et valeurs de constante (suite)	
Nom	Valeur
MQITS_NON_CONVERSATION	' - '
MQITS_ARCHITECTURE	' A '

**Remarque :** Le symbole - représente un caractère blanc unique.

### MQKAI\_\* (intervalleKeepAlive )

Tableau 212. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQKAI_AUTO	-1	X'FFFFFFFF'

### MQMASTER\_\* (administration maître)

Tableau 213. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQMASTER_NO	0	X'00000000'
MQMASTER_OUI	1	X'00000001'

### MQMCAS\_\* (Statut de l'agent de canal de message de format de commande)

Tableau 214. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQMCAS_ARRETA	0	X'00000000'
MQMCAS_EN COURS D'EXECUTION	3	X'00000003'

### MQMCAT\_\* (types MCA)

Tableau 215. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
PROCESSUS MQMCAT_PROCESS	1	X'00000001'
MQMCAT_THREAD	2	X'00000002'

### MQMCD\_\* (Informations sur la balise des options de publication / abonnement)

#### Options de publication / abonnement Balise Descripteur de contenu de message (mcd) Etiquettes

Tableau 216. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQMCD_FOLDER_VERSION	1	X'00000001'

#### Noms de balise d'options de publication / abonnement

Tableau 217. Noms et valeurs de constante	
Nom	Valeur
DOMAINE_MSG_MQMCD_MESSAGE	"Msd"

Tableau 217. Noms et valeurs de constante (suite)	
Nom	Valeur
MQMCD_MSG_SET	"Set"
TYPE MQMCD_MSG_TYPE	"Type"
FORMAT MQMCD_MSG_FORMAT	"Fmt"

### Noms de balise XML des options de publication / abonnement

Tableau 218. Noms et valeurs de constante	
Nom	Valeur
MQMCD_MSG_DOMAINE_B	"<Msd>"
MQMCD_MSG_DOMAINE_E	"</Msd>"
MQMCD_MSG_SET_B	"<Set>"
MQMCD_MSG_SET_E	"</Set>"
MQMCD_MSG_TYPE_B	"<Type>"
MQMCD_MSG_TYPE_E	"</Type>"
MQMCD_MSG_FORMAT_B	"<Fmt>"
MQMCD_MSG_FORMAT_E	"</Fmt>"

### Valeurs de balise d'options de publication / abonnement

Tableau 219. Noms et valeurs de constante	
Nom	Valeur
MQMCD_DOMAINE_AUCUN	"none"
MQMCD_DOMAIN_NEON	"neon"
MQMCD_DOMAIN_MRM	"mrm"
MQMCD_DOMAIN_JMS_NONE	"jms_none"
MQMCD_DOMAIN_JMS_TEXT	"jms_text"
OBJET_JMS_DOMAINE_MQMCD	"jms_object"
MAPPE_JMS_DOMAINE_MQMCD	"jms_map"
MQMCD_DOMAIN_JMS_STREAM	"jms_stream"
MQMCD_DOMAIN_JMS_BYTES	"jms_bytes"

### MQMD\_\* (structure de descripteur de message)

Tableau 220. Structures de constantes	
Nom	Structure
ID_STRUCD_MQM	"MD↵"
MQMD_STRUC_ID_ARRAY	'M', 'D', '↵', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 221. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQMD_VERSION_1	1	X'00000001'
MQMD_VERSION_2	2	X'00000002'

Tableau 221. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQMD_CURRENT_VERSION	2	X'00000002'

### MQMDE\_\* (structure d'extension de descripteur de message)

Tableau 222. Structures de constantes	
Nom	Structure
ID_STRUCDE_MQM	"MDE~"
MQMDE_STRUC_ID_ARRAY	'M', 'D', 'E', '~'

**Remarque :** Le symbole ~ représente un caractère blanc unique.

Tableau 223. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQMDE_VERSION_2	2	X'00000002'
MQMDE_CURRENT_VERSION	2	X'00000002'
MQMDE_LENGTH_2	72	X'00000048'

### MQMDEF\_\* (indicateurs d'extension de descripteur de message)

Tableau 224. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQMDEF_AUCUN	0	X'00000000'

### MQMDS\_\* (Séquence de distribution des messages)

Tableau 225. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQMDS_PRIORITE	0	X'00000000'
MQMDS_FIFO	1	X'00000001'

### MQMF\_\* (indicateurs de message)

Tableau 226. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQMF_SEGMENTATION_INHIBÉE	0	X'00000000'
MQMF_SEGMENTATION_ALLOWED	1	X'00000001'
GROUPE MQMF_MSG_IN_GROUP	8	X'00000008'
MQMF_LAST_MSG_IN_GROUP	16	X'00000010'
SEGMENT_MQMF	2	X'00000002'
SEGMENT_DERNIER_MQMF_SEGMENT	4	X'00000004'
MQMF_AUCUN	0	X'00000000'

## MQMHBO\_\* (descripteur de message vers les options et la structure de la mémoire tampon)

### Descripteur de message dans la structure des options de mémoire tampon

Tableau 227. Structures de constantes

Nom	Structure
MQMHBO_STRUC_ID	"MHBO"
MQMHBO_STRUC_ID_ARRAY	'M', 'H', 'B', 'O'

**Remarque :** Le symbole – représente un caractère blanc unique.

Tableau 228. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQMHBO_VERSION_1	1	X'00000001'
MQMHBO_CURRENT_VERSION	1	X'00000001'

### Options de traitement des messages vers la mémoire tampon

Tableau 229. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQMHBO_PROPERTIES_IN_MQRFH2	1	X'00000001'
MQMHBO_DELETE_PROPERTIES	2	X'00000002'
MQMHBO_AUCUN	0	X'00000000'

## MQMI\_\* (identificateur de message)

Tableau 230. Noms et valeurs de constante

Nom	Valeur
MQMI_AUCUN	X'00...00' (24 valeurs nulles)
MQMI_NON_ARRAY	'\0', '\0', ... (24 valeurs nulles)

## MQMMBI\_\* (Intervalle entre les marques de message)

Tableau 231. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQMMBI_UNLIMITED	-1	X'FFFFFFFF'

## MQMO\_\* (options de correspondance)

Tableau 232. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
ID MQMO_MATCH_MSG_ID	1	X'00000001'
ID_CORREL_MQMO_MATCH_	2	X'00000002'
ID_GROUPE_MQMO_MATCH	4	X'00000004'
NUMERO MQMO_MATCH_MSG_SEQ_NO	8	X'00000008'
MQMO_MATCH_OFFSET	16	X'00000010'
MQMO_MATCH_MSG_TOKEN	32	X'00000020'



Tableau 232. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQMO_AUCUN	0	X'00000000'

### MQMODE\_\* (Options de mode de format de commande)

Tableau 233. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
FORCE MQMODE	0	X'00000000'
MQMODE_QUIESCE	1	X'00000001'
MQMODE_TERMINATE	2	X'00000002'

### MQMON\_\* (Valeurs de surveillance)

Tableau 234. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQMON_NON DISPONIBLE	-1	X'FFFFFFFF'
MQMON_AUCUN	-1	X'FFFFFFFF'
MQMON_Q_DIR	-3	X'FFFFFFFD'
MQMON_OFF	0	X'00000000'
MQMON_ON	1	X'00000001'
MQMON_DISABLED	0	X'00000000'
MQMON_ENABLED	1	X'00000001'
MQMON_FAIBLE	17	X'00000011'
MQMON_MEDIUM	33	X'00000021'
MQMON_ELEVE	65	X'00000041'

### MQMT\_\* (types de message)

Tableau 235. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQMT_SYSTEM_FIRST	1	X'00000001'
MQMT_REQUEST	1	X'00000001'
MQMT_REPLY	2	X'00000002'
MQMT_DATAGRAM	8	X'00000008'
MQMT_REPORT	4	X'00000004'
MQMT_MQE_FIELDS_FROM_MQE	112	X'00000070'
MQMT_MQE_FIELDS (CHAMP_MQT)	113	X'00000071'
MQMT_SYSTEM_LAST	65535	X'0000FFFF'
MQMT_APPL_FIRST	65536	X'00010000'
MQMT_APPL_LAST	99999999	X'3B9AC9FF'

## MQMTOK\_\* (jeton de message)

Tableau 236. Noms et valeurs de constante	
Nom	Valeur
MQMTOK_NONE	X'00...00' (16 valeurs nulles)
MQMTOK_NON_ARRAY	'\0', '\0', ... (16 valeurs nulles)

## MQNC\_\* (nombre de noms)

Tableau 237. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQNC_MAX_NAMELIST_NAME_COUNT	256	X'00000100'

## MQNPM\_\* (classe de message non persistant)

Tableau 238. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQNPM_CLASS_NORMAL	0	X'00000000'
MQNPM_CLASS_HIGH	10	X'0000000A'

## MQNPMS\_\* (NonPersistent-Messages)

Tableau 239. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQNPMS_NORMAL	1	X'00000001'
MQNPMS_FAST	2	X'00000002'

## MQNT\_\* (Types de liste de noms)

Tableau 240. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQNT_AUCUN	0	X'00000000'
MQNT_Q	1	X'00000001'
MQNT_CLUSTER	2	X'00000002'
INFO MQNT_AUTH_INFO	4	X'00000004'
MQNT_ALL	1001	X'000003E9'

## MQNVS\_\* (Noms de la chaîne de nom / valeur)

Tableau 241. Noms et valeurs de constante	
Nom	Valeur
MQNVS_TYPE D'APPLICATION	"OPT_APP_GRP~"
TYPE MQNVS_MSG_	"OPT_MSG_TYPE~"

**Remarque :** Le symbole ~ représente un caractère blanc unique.

## MQOA\_\* (Limites des sélecteurs pour les attributs d'objet)

Tableau 242. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQOA_PREMIER	1	X'00000001'
MQOA_LAST	9000	X'00002328'

## MQOD\_\* (structure de descripteur d'objet)

Tableau 243. Structures de constantes	
Nom	Structure
ID_STRUC_MQODE	"0D↵"
MQOD_STRUC_ID_ARRAY	'0','D','↵','↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 244. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQOD_VERSION_1	1	X'00000001'
MQOD_VERSION_2	2	X'00000002'
MQOD_VERSION_3	3	X'00000003'
MQOD_VERSION_4	4	X'00000004'
VERSION MQOD_CURRENT_VERSION	4	X'00000004'
MQOD_LONGUEUR_ACTUELLE	(value differs by platform or version)	(value differs by platform or version)

## MQOII\_\* (identificateur d'instance d'objet)

Tableau 245. Noms et valeurs de constante	
Nom	Valeur
MQOII_AUCUN	X'00...00' (24 valeurs nulles)
MQOII_NONE_ARRAY	'\0','\0',... (24 valeurs nulles)

## MQOL\_\* (Longueur d'origine)

Tableau 246. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQOL_UNDEFINI	-1	X'FFFFFFFF'

## MQOM\_\* (options des messages Db2 obsolètes sur le groupe Inquire)

Tableau 247. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQOM_NO	0	X'00000000'
MQOM_OUI	1	X'00000001'

## MQOO\_\* (options d'ouverture)

Tableau 248. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQOO_BIND_AS_Q_DEF	0	X'00000000'
MQOO_READ_AHEAD_AS_Q_DEF	0	X'00000000'
MQOO_INPUT_AS_Q_DEF	1	X'00000001'
MQOO_INPUT_SHARED	2	X'00000002'
MQOO_INPUT_EXCLUSIVE	4	X'00000004'
MQOO_BROWSE	8	X'00000008'
MQOO_SORTIE	16	X'00000010'
MQOO_INTERROGATION	32	X'00000020'
MQOO_SET	64	X'00000040'
MQOO_SAVE_ALL_CONTEXT	128	X'00000080'
MQOO_PASS_IDENTITY_CONTEXT	256	X'00000100'
MQOO_PASS_ALL_CONTEXT	512	X'00000200'
MQOO_SET_IDENTITY_CONTEXT	1024	X'00000400'
MQOO_SET_ALL_CONTEXT	2048	X'00000800'
MQOO_ALTERNATE_USER_AUTHORITY	4096	X'00001000'
MQOO_FAIL_IF_QUIESCING	8192	X'00002000'
MQOO_BIND_ON_OPEN	16384	X'00004000'
MQOO_BIND_NOT_FIXED	32768	X'00008000'
MQOO_CO_OP	131072	X'00020000'
MQOO_RESOLVE_LOCAL_TOPIC	262144	X'00040000'
MQOO_NO_READ_AHEAD	524288	X'00080000'
MQOO_READ_AHEAD	1048576	X'00100000'
MQOO_BIND_ON_GROUPE	4194304	X'00400000'

## MQOO\_\* (suivi utilisé en C++ uniquement)

Tableau 249. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQOO_NOMS_RÉSOLUTION	65536	X'00010000'
MQOO_RÉSOLVE_LOCAL_Q	262144	X'00040000'

## MQOP\_\* (codes opération pour MQCTL et MQCB)

### Codes opération pour MQCTL

Tableau 250. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
DEMARRAGE MQOP	1	X'00000001'
MQOP_START_WAIT	2	X'00000002'
MQOP_ARRET	4	X'00000004'

## Codes opération pour MQCB

Tableau 251. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQOP_REGISTER	256	X'00000100'
MQOP_DÉSENREGISTREMENT	512	X'00000200'

## Codes opération pour MQCTL et MQCB

Tableau 252. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQOP_SUSPENSION	65536	X'00010000'
MQOP_RESUME	131072	X'00020000'

## MQOPEN\_\* (valeurs associées à la structure MQOPEN\_PRIV)

Tableau 253. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQOPEN_PRIV_VERSION_1	1	X'00000001'
MQOPEN_PRIV_CURRENT_VERSION	1	X'00000001'

## MQOPER\_\* (Opérations d'activité)

Tableau 254. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQOPER_SYSTEM_FIRST	0	X'00000000'
MQOPER_INCONNU	0	X'00000000'
MQOPER_PARCOURIR	1	X'00000001'
Carte MQOPER_DISCARD	2	X'00000002'
MQOPER_GET	3	X'00000003'
MQOPER_PUT	4	X'00000004'
MQOPER_PUT_REPLY	5	X'00000005'
RAPPORT MQOPER_PUT_REPORT	6	X'00000006'
MQOPER_RECEIVE	7	X'00000007'
MQOPER_SEND	8	X'00000008'
MQOPER_TRANSFORM	9	X'00000009'
MQOPER_PUBLICATION	10	X'0000000A'
MQOPER_EXCLUDED_PUBLISH	11	X'0000000B'
MQOPER_DISCARDED_PUBLISH	12	X'0000000C'
MQOPER_SYSTEM_LAST	65535	X'0000FFFF'
MQOPER_APPL_FIRST	65536	X'00010000'
MQOPER_APPL_LAST	99999999	X'3B9AC9FF'

## MQOT\_\* (types d'objet et types d'objet étendu)

### Types d'objet

Tableau 255. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQOT_AUCUN	0	X'00000000'
MQOT_Q	1	X'00000001'
MQOT_NAMELIST	2	X'00000002'
PROCESSUS MQ	3	X'00000003'
CLASSE DE STOCKAGE MQOT_STORAGE_CLASS	4	X'00000004'
MQOT_Q_DIR	5	X'00000005'
CANAL_MQTON	6	X'00000006'
INFO MQOT_AUTH_INFO	7	X'00000007'
MQOT_TOPIC	8	X'00000008'
MQOT_CF_STRUC	10	X'0000000A'
MQOT_PROGRAMME d'écoute	11	X'0000000B'
SERVICE MQOT	12	X'0000000C'
MQOT_RESERVED_1	999	X'000003E7'

### Types d'objet étendus

Tableau 256. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQOT_ALL	1001	X'000003E9'
MQOT_ALIAS_Q	1002	X'000003EA'
MQOT_MODEL_Q	1003	X'000003EB'
MQOT_LOCAL_Q	1004	X'000003EC'
MQOT_REMOTE_Q	1005	X'000003ED'
MQOT_SENDER_CHANNEL	1007	X'000003EF'
CANAL_SERVEUR_MQOT_CANAL	1008	X'000003F0'
CANAL_DEMANDE_MQOT_CANAL	1009	X'000003F1'
CANAL_RÉCEPTEUR_MQOT_CANAL	1010	X'000003F2'
MQOT_CANAL_EN_COURS	1011	X'000003F3'
CANAL_SAUVEGARDE_MQOT_SAVED_ENREGISTREMENT	1012	X'000003F4'
CANAL_SVRCONN_MQOT_SVRCONN_CHANNEL	1013	X'000003F5'
MQOT_CLNTCONN_CHANNEL	1014	X'000003F6'
MQOT_SHORT_CHANNEL	1015	X'000003F7'
MQOT_CHLAUTH	1016	X'000003F8'
MQOT_REMOTE_Q_MGR_NAME	1017	X'000003F9'
POLITIQUE MQOT_PROT_DE	1019	X'000003FB'
CANAL_TT_MQOT_TM	1020	X'000003FC'
CANAL_AMQP_MQOT_AMQP	1021	X'000003FD'

Tableau 256. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
RE_AUTH_MQOT_REC	1022	X'000003FE'

## MQPA\_\* (droits d'insertion)

Tableau 257. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQPA_PAR_DEFAULT	1	X'00000001'
MQPA_CONTEXT	2	X'00000002'
MQPA_ONLY_MCA	3	X'00000003'
MQPA_ALTERNATE_OR_MCA	4	X'00000004'

## MQPD\_\* (descripteur de propriété, support et contexte)

### Structure du descripteur de propriété

Tableau 258. Structures de constantes	
Nom	Structure
ID_STRUC_MQD	"PD↵"
MQPD_STRUC_ID_ARRAY	'P', 'D', '↵', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 259. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQPD_VERSION_1	1	X'00000001'
MQPD_CURRENT_VERSION	1	X'00000001'

### Options de descripteur de propriété

Tableau 260. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQPD_AUCUN	0	X'00000000'

### Options de prise en charge des propriétés

Tableau 261. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQPD_SUPPORT_XX_ENCODE_CASE_ONE facultatif	1	X'00000001'
MQPD_SUPPORT_REQUIS	1048576	X'00100000'
MQPD_SUPPORT_REQUIRED_IF_LOCAL	1024	X'00000400'
MQPD_REJECT_UNSUP_MASK	-1048576	X'FFF00000'
MQPD_ACCEPT_UNSUP_IF_XMIT_MASK	1047552	X'000FFC00'
MQPD_ACCEPT_UNSUP_MASK	1023	X'000003FF'

## Contexte de propriété

Tableau 262. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQPD_NO_CONTEXT	0	X'00000000'
CONTEXTE MQPD_USER_	1	X'00000001'

## MQPER\_\* (Valeurs de persistance)

Tableau 263. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQPER_PERSISTENCE_AS_PARENT	-1	X'FFFFFFFF'
MQPER_NON_PERSISTENT	0	X'00000000'
MQPER_PERSISTANT	1	X'00000001'
MQPER_PERSISTENCE_AS_Q_DEF	2	X'00000002'
MQPER_PERSISTENCE_AS_TOPIC_DEF	2	X'00000002'

## MQPL\_\* (plateformes)

Tableau 264. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQPL_MVS	1	X'00000001'
MQPL_OS390	1	X'00000001'
MQPL_ZOS	1	X'00000001'
MQPL_OS2	2	X'00000002'
MQPL_AIX	3	X'00000003'
MQPL_UNIX	3	X'00000003'
MQPL_OS400	4	X'00000004'
MQPL_WINDOWS	5	X'00000005'
MQPL_WINDOWS_NT	11	X'0000000B'
MQPL_VMS	12	X'0000000C'
MQPL_NSK	13	X'0000000D'
MQPL_OPEN_TP1	15	X'0000000F'
MQPL_MV	18	X'00000012'
MQPL_TPF	23	X'00000017'
MQPL_VSE	27	X'0000001B'
MQPL_DISPOSITIF	28	X'0000001C'
MQPL_NATIVE	1	X'00000001'



## MQPMO\_\* (Options de message d'insertion et structure pour le masque de publication)

### Structure des options de message d'insertion

Tableau 265. Structures de constantes

Nom	Structure
ID_STRUC_MQPMO_	"PMO~"
MQPMO_STRUC_ID_ARRAY	'P', 'M', 'O', '~'

**Remarque :** Le symbole ~ représente un caractère blanc unique.

Tableau 266. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQPMO_VERSION_1	1	X'00000001'
MQPMO_VERSION_2	2	X'00000002'
MQPMO_VERSION_3	3	X'00000003'
MQPMO_CURRENT_VERSION	3	X'00000003'
MQPMO_LONGUEUR_ACTUELLE	(value differs by platform or version)	(value differs by platform or version)

### Options d'insertion de message

Tableau 267. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQPMO_SYNCPOINT	2	X'00000002'
MQPMO_NO_SYNCPOINT	4	X'00000004'
MQPMO_CONTEXTE_PAR_DÉFAUT	32	X'00000020'
ID MQPMO_NEW_MSG_ID	64	X'00000040'
MQPMO_NEW_CORREL_ID	128	X'00000080'
MQPMO_PASS_IDENTITY_CONTEXT	256	X'00000100'
MQPMO_PASS_ALL_CONTEXT	512	X'00000200'
MQPMO_SET_IDENTITY_CONTEXT	1024	X'00000400'
MQPMO_SET_ALL_CONTEXT	2048	X'00000800'
MQPMO_ALTERNATE_USER_AUTHORITY	4096	X'00001000'
MQPMO_FAIL_IF QUIESCING	8192	X'00002000'
MQPMO_NO_CONTEXT	16384	X'00004000'
MQPMO_LOGICAL_ORDER	32768	X'00008000'
MQPMO_ASYNC_RESPONSE	65536	X'00010000'
MQPMO_SYNC_REPONSE	131072	X'00020000'
MQPMO_RESOLVE_LOCAL_Q	262144	X'00040000'
MQPMO_RETAIN	2097152	X'00200000'
MQPMO_MD_FOR_OUTPUT_ONLY	8388608	X'00800000'
MQPMO_SCOPE_QMGR	67108864	X'04000000'
MQPMO_SUPPRESS_REPLYTO	134217728	X'08000000'

Tableau 267. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQPMO_NOT_OWN_SUBS	268435456	X'10000000'
MQPMO_REPONSE_AS_Q_DEF	0	X'00000000'
MQPMO_REPONSE_AS_TOPIC_DEF	0	X'00000000'
MQPMO_NONE	0	X'00000000'

### Options d'insertion de message pour le masque de publication

Tableau 268. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQPMO_PUB_OPTIONS_MASK	2097152	X'00200000'

### MQPMRF\_\* (Zones d'enregistrement de message d'insertion)

Tableau 269. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
ID MQPMRF_MSG_ID	1	X'00000001'
ID_CORREL_MQPMRF	2	X'00000002'
ID_GROUPE_MQPMRF_ID	4	X'00000004'
MQPMRF_FEEDBACK	8	X'00000008'
MQPMRF_COMPTING_TOKEN	16	X'00000010'
MQPMRF_AUCUN	0	X'00000000'

### MQPO\_\* (options de purge de format de commande)

Tableau 270. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQPO_OUI	1	X'00000001'
MQPO_NON	0	X'00000000'

### MQPRI\_\* (priorité)

Tableau 271. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQPRI_PRIORITY_AS_Q_DEF	-1	X'FFFFFFFF'
MQPRI_PRIORITY_AS_PARENT	-2	X'FFFFFFFE'
MQPRI_PRIORITY_AS_PUBLISHED	-3	X'FFFFFFFD'
MQPRI_PRIORITY_AS_TOPIC_DEF	-1	X'FFFFFFFF'

## MQPROP\_\* (Valeurs de contrôle de propriété de file d'attente et de canal et longueur maximale des propriétés)

### Valeurs de contrôle de propriété de file d'attente et de canal

Tableau 272. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
COMPATIBILITE_MQPROP_COMPATIBILITÉ	0	X'00000000'
MQPROP_NONE	1	X'00000001'
MQPROP_ALL	2	X'00000002'
MQPROP_FORCE_MQRFH2	3	X'00000003'

### Longueur maximale des propriétés

Tableau 273. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQPROP_UNRESTRICTED_LENGTH	-1	X'FFFFFFFF'

## MQPRT\_\* (Valeurs de réponse d'insertion)

Tableau 274. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQPRT_RESPONSE_AS_PARENT	0	X'00000000'
MQPRT_SYNC_REPONSE	1	X'00000001'
MQPRT_ASYNC_RESPONSE	2	X'00000002'

## MQPS\_\* (Publier/S' abonner)

### Statut de publication / d'abonnement du format de commande

Tableau 275. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQPS_STATUS_INACTIVE	0	X'00000000'
MQPS_STATUS_DEMARRAGE	1	X'00000001'
MQPS_STATUS_STOPPING	2	X'00000002'
STATUS_MQPS ACTIF	3	X'00000003'
MQPS_STATUS_COMPAT	4	X'00000004'
MQPS_STATUS_ERREUR	5	X'00000005'
MQPS_STATUS_REFUSE	6	X'00000006'

### Balises de publication / abonnement en tant que chaînes

COMMANDE MQPS	"MQPSCommand"
CODE COMP_MQPS	"MQPSCompCode"
ID_CORREL_MQPS	"MQPSCorrelId"
OPTIONS DE DELETE_MQPS	"MQPSDe10pts"

ID_ERREUR_MQPS	"MQPSErrorId"
MQPS_ERREUR_POS	"MQPSErrorPos"
DONNEES_INTEGER_MQPS	"MQPSIntData"
MQPS_PARAMETER_ID	"MQPSParmId"
OPTIONS DE PUBLICATION_MQPS	"MQPSPubOpts"
MQPS_PUBLISH_TIMESTAMP	"MQPSPubTime"
MQPS_Q_MGR_NAME	"MQPSQMgrName"
NOM_Q_MQPS	"MQPSQName"
MQPS_MOTIF	"MQPSReason"
MQPS_MOTIF	"MQPSReasonText"
OPTIONS D'ENREGISTREMENT MQPS	"MQPSRegOpts"
NUMÉRO_SÉQUENCE_MQPS_	"MQPSSeqNum"
NOM_FLOT_MQPS_	"MQPSStreamName"
DONNEES_STRING_MQPS	"MQPSStringData"
MQPS_SUBSCRIPTION_IDENTITY	"MQPSSubIdentity"
NOM_SUBSCRIPTION_MQPS	"MQPSSubName"
MQPS_SUBSCRIPTION_USER_DATA	"MQPSSubUserData"
MQPS_TOPIC	"MQPSTopic"
ID_UTILISATEUR_MQPS	"MQPSUserId"

### Balises de publication / abonnement sous forme de chaînes vides incluses

MQPS_COMMAND_B	"-MQPSCommand-"
MQPS_COMP_CODE_B	"-MQPSCompCode-"
MQPS_CORREL_ID_B	"-MQPSCorrelId-"
MQPS_DELETE_OPTIONS_B	"-MQPSDelOpts-"
ID_ERREUR_MQPS B	"-MQPSErrorId-"
MQPS_ERRE_POS_B	"-MQPSErrorPos-"
MQPS_INTEGER_DATA_B	"-MQPSIntData-"
MQPS_PARAMETER_ID_B	"-MQPSParmId-"
MQPS_PUBLICATION_OPTIONS_B	"-MQPSPubOpts-"
MQPS_PUBLISH_HORODATAGE_B	"-MQPSPubTime-"
MQPS_Q_MGR_NAME_B	"-MQPSQMgrName-"
MQPS_Q_NOM_B	"-MQPSQName-"
MQPS_MOTIF B	"-MQPSReason-"
MQPS_REASON_TEXT_B	"-MQPSReasonText-"
MQPS_REGISTRATION_OPTIONS_B	"-MQPSRegOpts-"

NOMBRE_SÉQUENCE_MQPS_B	"¬MQPSSeqNum¬"
MQPS_STREAM_NAME_B	"¬MQPSStreamName¬"
MQPS_STRING_DATA_B	"¬MQPSStringData¬"
MQPS_SUBSCRIPTION_IDENTITY_B	"¬MQPSSubIdentity¬"
MQPS_SUBSCRIPTION_NAME_B	"¬MQPSSubName¬"
MQPS_SUBSCRIPTION_USER_DATA_B	"¬MQPSSubUserData¬"
MQPS_TOPIC_B	"¬MQPSTopic¬"
ID_utilisateur_MQPS_B	"¬MQPSUserId¬"

**Remarque :** Le symbole ¬ représente un caractère blanc unique.

### Valeurs de balise de commande de publication / abonnement sous forme de chaînes

MQPS_DELETE_PUBLICATION	"DeletePub"
MQPS_DÉSENREGISTRE_PUBLIEUR	"DeregPub"
ABONNEMENT_DÉSENREGISTRE_MQP	"DeregSub"
MQPS_PUBLICATION	"Publish"
MQPS_REGISTER_DIFFUSEUR de publications	"RegPub"
ABONNÉ_REGISTER_MQPS_	"RegSub"
MQPS_REQUEST_UPDATE	"ReqUpdate"

### Valeurs de balise de commande de publication / abonnement sous forme de chaînes encadrées par des blancs

MQPS_DELETE_PUBLICATION_B	"¬DeletePub¬"
MQPS_DÉSENREGISTRE_PUBLISHER_B	"¬DeregPub¬"
MQPS_DEREGISTER_SUBSCRIBER_B	"¬DeregSub¬"
MQPS_PUBLISH_B	"¬Publish¬"
MQPS_REGISTER_PUBLISHER_B	"¬RegPub¬"
MQPS_REGISTER_SUBSCRIBER_B	"¬RegSub¬"
MQPS_REQUEST_UPDATE_B	"¬ReqUpdate¬"

**Remarque :** Le symbole ¬ représente un caractère blanc unique.

### Valeurs de balise d'options de publication / abonnement sous forme de chaînes

NOM_ADD_MQ	"AddName"
MQPS_ANONYME	"Anon"
MQPS_CORREL_ID_AS_IDENTITE	"CorrelAsId"
MQPS_DEREGISTER_ALL	"DeregAll"
MQPS_DIRECT_REQUESTS	"DirectReq"
MQPS_DUPLICATES_OK	"DupsOK"

MQPS_RÉPONSE_COMPLET_RÉPONSE	"FullResp"
MQPS_INCLUDE_STREAM_NAME	"InclStreamName"
MQPS_INFORM_SI_RETENU	"InformIfRet"
MQPS_IS_RETAINED_PUBLICATION	"IsRetainedPub"
MQPS_JOIN_EXCLUSIF	"JoinExcl"
MQPS_JOIN_SHARED	"JoinShared"
MQPS_LEAVE_ONLY	"LeaveOnly"
MQPS_LOCAL	"Local"
MQPS_VERROUILLÉ	"Locked"
MQPS_NEW_PUBLICATIONS_ONLY	"NewPubsOnly"
MQPS_NON_MODIFICATION	"NoAlter"
MQPS_NO_REGISTRATION	"NoReg"
MQPS_NON_PERSISTENT	"NonPers"
MQPS_AUCUN	"None"
MQPS_AUTRE_ABONNEMENTS uniquement	"OtherSubsOnly"
MQPS_PERSISTENT	"Pers"
MQPS_PERSISTENT_AS_PUBLISH	"PersAsPub"
MQPS_XX_ENCODE_CASE_ONE persistant-AS_Q	"PersAsQueue"
MQPS_PUBLISH_ON_REQUEST_ONLY	"PubOnReqOnly"
PUBLICATION MQPS_RETAIN_	"RetainPub"
ID_UTILISATEUR_MQPS_VARIABLE_	"VariableUserId"

**Valeurs de balise d'options de publication / d'abonnement sous forme de chaînes entourées de blancs**

MQPS_ADD_NAME_B	"-AddName-"
MQPS_ANONYMOUS_B	"-Anon-"
MQPS_CORREL_ID_AS_IDENTITY_B	"-CorrelAsId-"
MQPS_DEREGISTER_ALL_B	"-DeregAll-"
MQPS_DIRECT_REQUESTS_B	"-DirectReq-"
MQPS_DUPLICES_OK_B	"-DupsOK-"
MQPS_FULL_RESPONSE_B	"-FullResp-"
MQPS_INCLUDE_STREAM_NAME_B	"-InclStreamName-"
MQPS_INFORM_IF_RETAINED_B	"-InformIfRet-"
MQPS_IS_RETAINED_PUBLICATION_B	"-IsRetainedPub-"
MQPS_JOIN_EXCLUSIVE_B	"-JoinExcl-"
MQPS_JOIN_SHARED_B	"-JoinShared-"
MQPS_LEAVE_ONLY_B	"-LeaveOnly-"

MQPS_LOCAL_B	"¬Local¬"
MQPS_LOCKED_B	"¬Locked¬"
MQPS_NEW_PUBLICATIONS_ONLY_B	"¬NewPubsOnly¬"
MQPS_NO_ALTERATION_B	"¬NoAlter¬"
MQPS_NO_REGISTRATION_B	"¬NoReg¬"
MQPS_NON_PERSISTENT_B	"¬NonPers¬"
MQPS_NONE_B	"¬None¬"
MQPS_AUTRE_ABONNEMENTS_ONLY_B	"¬OtherSubsOnly¬"
MQPS_XX_ENCODE_CASE_ONE persistant _b	"¬Pers¬"
MQPS_PERSISTENT_AS_PUBLISH_B	"¬PersAsPub¬"
MQPS_XX_ENCODE_CASE_ONE persistant AS_q_b	"¬PersAsQueue¬"
MQPS_PUBLISH_ON_REQUEST_ONLY_B	"¬PubOnReqOnly¬"
MQPS_RETAIN_PUBLICATION_B	"¬RetainPub¬"
ID_UTILISATEUR_MQPS_VARIABLE_B	"¬VariableUserId¬"

**Remarque :** Le symbole ¬ représente un caractère blanc unique.

### **MQPSC\_\* (balises d'options de publication / d'abonnement-Dossier de commandes de publication / d'abonnement (psc))**

<i>Tableau 276. Valeurs des constantes</i>		
Nom	Valeur décimale	Valeur hexadécimale
MQPSC_FOLDER_VERSION	1	X'00000001'

### **MQPSC\_\* (noms de balise d'options de publication / abonnement)**

MQPSC_COMMAND	"Command"
OPTION D'ENREGISTREMENT MQPS	"RegOpt"
OPTION MQPSC_PUBLICATION_OPTION	"PubOpt"
OPTION DELETE_MQPS	"DelOpt"
MQPSC_TOPIC	"Topic"
MQPSC_SUBSCRIPTION_POINT	"SubPoint"
MQPSC_FILTER	"Filter"
MQPSC_Q_MGR_NAME	"QMgrName"
MQPSC_Q_NAME	"QName"
MQPSC_HORODATAGE de publication	"PubTime"
NUMERO_SEQUENCE_MQPS	"SeqNum"
NOM_SOUS-SCRIPTION_MQPS	"SubName"
MQPSC_SUBSCRIPTION_IDENTITY	"SubIdentity"
MQPSC_SUBSCRIPTION_DONNEES_UTILISATEUR	"SubUserData"

ID_CORREL_MQPS	"CorrelId"
----------------	------------

***MQPSC\_\* (noms de balise XML des options de publication / abonnement)***

MQPSC_COMMAND_B	"<Command>"
MQPSC_COMMAND_E	"</Command>"
MQPSC_REGISTRATION_OPTION_B	"<RegOpt>"
MQPSC_REGISTRATION_OPTION_E	"</RegOpt>"
MQPSC_PUBLICATION_OPTION_B	"<PubOpt>"
MQPSC_PUBLICATION_OPTION_E	"</PubOpt>"
MQPSC_DELETE_OPTION_B	"<DelOpt>"
MQPSC_DELETE_OPTION_E	"</DelOpt>"
MQPSC_TOPIC_B	"<Topic>"
MQPSC_TOPIC_E	"</Topic>"
MQPSC_SUBSCRIPTION_POINT_B	"<SubPoint>"
MQPSC_SUBSCRIPTION_POINT_E	"</SubPoint>"
MQPSC_FILTER_B	"<Filter>"
MQPSC_FILTER_E	"</Filter>"
MQPSC_Q_MGR_NAME_B	"<QMgrName>"
MQPSC_Q_MGR_NAME_E	"</QMgrName>"
MQPSC_Q_NAME_B	"<QName>"
MQPSC_Q_NAME_E	"</QName>"
MQPSC_PUBLISH_HORODATAGE_B	"<PubTime>"
MQPSC_PUBLISH_TIMESTAMP_E	"</PubTime>"
MQPSC_NUMÉRO_SÉQUENCE_B	"<SeqNum>"
MQPSC_NUMÉRO_SÉQUENCE_E	"</SeqNum>"
MQPSC_SUBSCRIPTION_NAME_B	"<SubName>"
MQPSC_SUBSCRIPTION_NAME_E	"</SubName>"
MQPSC_SUBSCRIPTION_IDENTITY_B	"<SubIdentity>"
MQPSC_SUBSCRIPTION_IDENTITY_E	"</SubIdentity>"
MQPSC_SUBSCRIPTION_USER_DATA_B	"<SubUserData>"
MQPSC_SUBSCRIPTION_USER_DATA_E	"</SubUserData>"
MQPSC_CORREL_ID_B	"<CorrelId>"
ID_CORREL_MQPS E	"</CorrelId>"

***MQPSC\_\* (Options de publication / d'abonnement-Valeurs de diffuseur de publications sous forme de chaînes)***

MQPSC_DELETE_PUBLICATION	"DeletePub"
--------------------------	-------------



MQPSC_DÉSENREGISTRE_ABONNÉ	"DeregSub"
MQPSC_PUBLISH	"Publish"
MQPSC_REGISTER_ABONNÉ	"RegSub"
MQPSC_REQUEST_UPDATE	"ReqUpdate"

**MQPSC\_\* (Valeurs de nom de balise d'options de publication / abonnement sous forme de chaînes)**

NOM_ADD_MQPS	"AddName"
MQPSC_CORREL_ID_AS_IDENTITY	"CorrelAsId"
MQPSC_DEREGISTER_ALL	"DeregAll"
MQPSC_DUPLICATES_OK	"DupsOK"
MQPSC_FULL_RESPONSE	"FullResp"
MQPSC_INFORM_IF_RETAINED	"InformIfRet"
MQPSC_IS_RETAINED_PUB	"IsRetainedPub"
MQPSC_JOIN_SHARED	"JoinShared"
MQPSC_JOIN_EXCLUSIVE	"JoinExcl"
MQPSC_LEAVE_ONLY	"LeaveOnly"
MQPSC_LOCAL	"Local"
MQPSC_LOCKED	"Locked"
MQPSC_NEW_PUBS_ONLY	"NewPubsOnly"
MQPSC_NO_ALTERATION	"NoAlter"
MQPSC_NON_PERSISTENT	"NonPers"
MQPSC_OTHER_SUBS_ONLY	"OtherSubsOnly"
MQPSC_XX_ENCODE_CASE_ONE persistant	"Pers"
MQPSC_PERSISTENT_AS_PUBLISH	"PersAsPub"
MQPSC_XX_ENCODE_CASE_ONE persistENT_AS_q	"PersAsQueue"
MQPSC_NONE	"None"
MQPSC_PUB_ON_REQUEST_ONLY	"PubOnReqOnly"
MQPSC_RETAIN_PUB	"RetainPub"
ID_UTILISATEUR MQPSC_VARIABLE_	"VariableUserId"

**MQPSCR\_\* (options de publication / abonnement)**

**Options de publication / abonnement Etiquette Dossier de réponses de publication / abonnement (pscr) Etiquettes**

Tableau 277. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQPSCR_FOLDER_VERSION	1	X'00000001'

## Noms de balise d'options de publication / abonnement

MQPSCR_COMPLÉTION	"Completion"
MQPSCR_REPONSE	"Response"
MQPSCR_MOTIF	"Reason"

## Noms de balise XML des options de publication / abonnement

MQPSCR_COMPLETION_B	"<Completion>"
MQPSCR_COMPLETION_E	"</Completion>"
REPONSE_MQPSCR_B	"<Response>"
REPONSE_MQPSCR_E	"</Response>"
MQPSCR_MOTIF B	"<Reason>"
MQPSCR_MOTIF	"</Reason>"

## Valeurs de balise d'options de publication / abonnement

MQPSCR_OK	"ok"
AVERTISSEMENT MQPSCR_avis	"warning"
MQPSCR_ERREUR	"error"

## MQPSM\_\* (mode de publication / abonnement)

Tableau 278. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQPSM_DISABLED	0	X'00000000'
MQPSM_COMPAT	1	X'00000001'
MQPSM_ENABLED	2	X'00000002'

## MQPSPROP\_\* (Propriétés de message de publication / abonnement)

Tableau 279. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQPSPROP_NONE	0	X'00000000'
MQPSPROP_COMPAT	1	X'00000001'
MQPSPROP_RFH2	2	X'00000002'
MQPSPROP_MSGPROP	3	X'00000003'

## MQPSST\_\* (format de commande-Type de statut de publication / d'abonnement)

Tableau 280. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQPSST_ALL	0	X'00000000'
MQPSST_LOCAL	1	X'00000001'
MQPSST_PARENT	2	X'00000002'

Tableau 280. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQPSST_CHILD	3	X'00000003'

### MQPUBO\_\* (Options de publication / abonnement)

Tableau 281. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQPUBO_NONE	0	X'00000000'
MQPUBO_CORREL_ID_AS_IDENTITY	1	X'00000001'
MQPUBO_RETAIN_PUBLICATION	2	X'00000002'
MQPUBO_OTHER_SUBSCRIBERS_ONLY	4	X'00000004'
MQPUBO_NO_REGISTRATION	8	X'00000008'
MQPUBO_IS_RETAINED_PUBLICATION	16	X'00000010'

### MQPXP\_\* (structure des paramètres d'exit de routage de publication / abonnement)

Tableau 282. Structures de constantes	
Nom	Structure
MQPXP_STRUC_ID	"PXP↵"
MQPXP_STRUC_ID_ARRAY	'P','X','P','↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 283. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQPXP_VERSION_1	1	X'00000001'
MQPXP_CURRENT_VERSION	1	X'00000001'

### MQQA\_\* (attributs de file d'attente)

#### Interdire l'obtention de valeurs

Tableau 284. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQQA_GET_INHIBÉE	1	X'00000001'
MQQA_GET_ALLOWED	0	X'00000000'

#### Interdire les valeurs d'insertion

Tableau 285. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQQA_PUT_INHIBÉ	1	X'00000001'
MQQA_PUT_ALLOWED	0	X'00000000'

## Partage de file d'attente

Tableau 286. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQQA_SHAREABLE	1	X'00000001'
MQQA_NOT_SHAREABLE	0	X'00000000'

## Renforcement de l'arrière-système

Tableau 287. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQQA_BACKOUT_HARDENED	1	X'00000001'
MQQA_BACKOUT_NOT_HARDENED	0	X'00000000'

## MQQDT\_\* (types de définition de file d'attente)

Tableau 288. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQQDT_PRÉDÉFINI	1	X'00000001'
MQQDT_PERMANENT_DYNAMIQUE	2	X'00000002'
MQQDT_TEMPORARY_DYNAMIQUE	3	X'00000003'
MQQDT_PARTAGE_DYNAMIQUE	4	X'00000004'

## MQQF\_\* (indicateurs de file d'attente)

Tableau 289. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQQF_LOCAL_Q	1	X'00000001'
MQQF_CLWL_USEQ_ANY	64	X'00000040'
MQQF_CLWL_USEQ_LOCAL	128	X'00000080'

## MQQMDT\_\* (format de commande-Types de définition de gestionnaire de files d'attente)

Tableau 290. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQQMDT_EXPLICIT_CLUSTER_SENDER	1	X'00000001'
MQQMDT_AUTO_CLUSTER_SENDER	2	X'00000002'
MQQMDT_AUTO_EXP_CLUSTER_SENDER	4	X'00000004'
MQQMDT_CLUSTER_RECEIVER	3	X'00000003'

## MQQMF\_\* (indicateurs de gestionnaire de files d'attente)

Tableau 291. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQQMF_REPOSITORY_Q_MGR	2	X'00000002'
MQQMF_CLUSSDR_USER_DÉFINI	8	X'00000008'

Tableau 291. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQQMF_CLUSSDR_AUTO_DEFINED	16	X'00000010'
MQQMF_XX_ENCODE_CASE_ONE disponible	32	X'00000020'

### MQQMFACT\_\* (fonction de gestionnaire de files d'attente de format de commande)

Tableau 292. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQQMFACT_IMS_BRIDGE	1	X'00000001'
MQQMFACT_DB2	2	X'00000002'

### MQQMSTA\_\* (statut du gestionnaire de files d'attente de format de commande)

Tableau 293. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQQMSTA_DEMARRAGE	1	X'00000001'
MQQMSTA_EN COURS D'EXECUTION	2	X'00000002'
MQQMSTA_MISE au repos	3	X'00000003'

### MQQMT\_\* (Types de gestionnaire de files d'attente de format de commande)

Tableau 294. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQQMT_NORMAL	0	X'00000000'
MQQMT_REPOSITORY	1	X'00000001'

### MQQO\_\* (Options de mise au repos du format de commande)

Tableau 295. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQQO_OUI	1	X'00000001'
MQQO_NON	0	X'00000000'

### MQQSGD\_\* (dispositions de groupe de partage de files d'attente)

Tableau 296. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQQSGD_ALL	-1	X'FFFFFFFF'
MQQSGD_Q_DIR	0	X'00000000'
MQQSGD_COPY	1	X'00000001'
MQQSGD_SHARED	2	X'00000002'
GROUPE_MQQSG	3	X'00000003'
MQQSGD_PRIVÉ	4	X'00000004'
MQQSGD_LIVE	6	X'00000006'

## MQQSGS\_\* (statut du groupe de partage de files d'attente de format de commande)

Tableau 297. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQQSGS_INCONNU	0	X'00000000'
MQQSGS_CREE	1	X'00000001'
MQQSGS_ACTIF	2	X'00000002'
MQQSGS_XX_ENCODE_CASE_ONE inactif	3	X'00000003'
MQQSGS_ECHEC	4	X'00000004'
MQQSGS_PENDING	5	X'00000005'

## MQQSIE\_\* (Service de file d'attente de format de commande-Evénements d'intervalle)

Tableau 298. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQQSIE_NONE	0	X'00000000'
MQQSIE_HIGH	1	X'00000001'
MQQSIE_OK	2	X'00000002'

## MQQSO\_\* (options d'ouverture de statut de file d'attente de format de commande pour SET, BROWSE, INPUT)

Tableau 299. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQQSO_NO	0	X'00000000'
MQQSO_OUI	1	X'00000001'
MQQSO_SHARED	1	X'00000001'
MQQSO_EXCLUSIF	2	X'00000002'

## MQQSOT\_\* (Types d'ouverture de statut de file d'attente de format de commande)

Tableau 300. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQQSOT_ALL	1	X'00000001'
MQQSOT_ENTREE	2	X'00000002'
MQQSOT_OUTPUT	3	X'00000003'

## MQQSUM\_\* (format de commande-Messages de statut de file d'attente non validés)

Tableau 301. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQQSUM_OUI	1	X'00000001'
MQQSUM_NO	0	X'00000000'

## MQQT\_\* (types de file d'attente et types de file d'attente étendue)

### Types de file d'attente

Tableau 302. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQQT_LOCAL	1	X'00000001'
MQQT_MODELE	2	X'00000002'
MQQT_ALIAS (alias MQ)	3	X'00000003'
MQQT_REMOTE	6	X'00000006'
MQQT_CLUSTER	7	X'00000007'

### Types de file d'attente étendue

Tableau 303. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQQT_ALL	1001	X'000003E9'

### MQRC\_\* (codes anomalie)

Tableau 304. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRC_NONE	0	X'00000000'
MQRC_APPL_FIRST	900	X'00000384'
MQRC_APPL_LAST	999	X'000003E7'
MQRC_ALIAS_BASE_Q_TYPE_ERROR	2001	X'000007D1'
MQRC_ALREADY_CONNECTED	2002	X'000007D2'
MQRC_BACKED_OUT	2003	X'000007D3'
MQRC_BUFFER_ERROR	2004	X'000007D4'
MQRC_BUFFER_LENGTH_ERROR	2005	X'000007D5'
MQRC_CHAR_ATTR_LENGTH_ERROR	2006	X'000007D6'
MQRC_CHAR_ATTRS_ERREUR	2007	X'000007D7'
MQRC_CHAR_ATTRS_TOO_SHORT	2008	X'000007D8'
MQRC_CONNECTION_BROKEN	2009	X'000007D9'
MQRC_DATA_LENGTH_ERROR	2010	X'000007DA'
MQRC_DYNAMIC_Q_NAME_ERROR	2011	X'000007DB'
MQRC_ENVIRONMENT_ERROR	2012	X'000007DC'
MQRC_EXPIRY_ERROR	2013	X'000007DD'
MQRC_ERREUR_FEEDBACK_ERROR	2014	X'000007DE'
MQRC_GET_INHIBITED	2016	X'000007E0'
MQRC_HANDLE_NOT_AVAILABLE	2017	X'000007E1'
MQRC_HCONN_ERROR	2018	X'000007E2'
MQRC_HOBJ_ERROR	2019	X'000007E3'
MQRC_INHIBIT_VALEUR_ERREUR	2020	X'000007E4'

Tableau 304. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_INT_ATTR_COUNT_ERROR	2021	X'000007E5'
MQRC_INT_ATTR_COUNT_TOO_SMALL	2022	X'000007E6'
MQRC_INT_ATTRS_ARRAY_ERREUR	2023	X'000007E7'
MQRC_SYNCPOINT_LIMIT_REACHED	2024	X'000007E8'
MQRC_MAX_CONNS_LIMIT_ATTEINTE	2025	X'000007E9'
MQRC_MD_ERROR	2026	X'000007EA'
MQRC_MISSING_REPLY_TO_Q	2027	X'000007EB'
MQRC_MSG_TYPE_ERREUR	2029	X'000007ED'
MQRC_MSG_TOO_BIG_FOR_Q	2030	X'000007EE'
MQRC_MSG_TOO_BIG_FOR_Q_MGR	2031	X'000007EF'
MQRC_NO_MSG_AVAILABLE	2033	X'000007F1'
MQRC_NO_MSG_UNDER_CURSOR	2034	X'000007F2'
MQRC_NOT_AUTHORIZED	2035	X'000007F3'
MQRC_NOT_OPEN_FOR_BROWSE	2036	X'000007F4'
MQRC_NOT_OPEN_FOR_INPUT	2037	X'000007F5'
MQRC_NOT_OPEN_FOR_INQUIRE	2038	X'000007F6'
MQRC_NOT_OPEN_FOR_OUTPUT	2039	X'000007F7'
MQRC_NOT_OPEN_FOR_SET	2040	X'000007F8'
MQRC_OBJECT_CHANGED	2041	X'000007F9'
MQRC_OBJECT_IN_USE	2042	X'000007FA'
MQRC_OBJECT_TYPE_ERREUR	2043	X'000007FB'
MQRC_ERREUR D'OD_DU	2044	X'000007FC'
MQRC_OPTION_NOT_VALID_FOR_TYPE	2045	X'000007FD'
MQRC_OPTIONS_ERROR	2046	X'000007FE'
MQRC_XX_ENCODE_CASE_ONE err_persistent	2047	X'000007FF'
MQRC_PERSISTENT_NOT_ALLOWED	2048	X'00000800'
MQRC_PRIORITY_MAXIMAL_DÉPASSEMENTS	2049	X'00000801'
MQRC_PRIORITY_ERROR	2050	X'00000802'
MQRC_PUT_INHIBÉ	2051	X'00000803'
MQRC_Q_DELETED	2052	X'00000804'
MQRC_Q_FULL	2053	X'00000805'
MQRC_Q_NOT_EMPTY	2055	X'00000807'
MQRC_Q_ESPACE_NON_DISPONIBLE	2056	X'00000808'
MQRC_Q_TYPE_ERREUR	2057	X'00000809'
MQRC_Q_MGR_NAME_ERROR	2058	X'0000080A'
MQRC_Q_MGR_NOT_AVAILABLE	2059	X'0000080B'
MQRC_REPORT_OPTIONS_ERREUR	2061	X'0000080D'
MQRC_SECOND_MARK_NOT_ALLOWED	2062	X'0000080E'
MQRC_SECURITY_ERROR (ERREUR DE SECURITE MQ)	2063	X'0000080F'



Tableau 304. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_SELECTOR_COUNT_ERREUR	2065	X'00000811'
MQRC_SELECTOR_LIMIT_EXCEEDED	2066	X'00000812'
MQRC_SELECTOR_ERREUR	2067	X'00000813'
MQRC_SELECTOR_NOT_FOR_TYPE	2068	X'00000814'
MQRC_SIGNAL_OUTSTANDING	2069	X'00000815'
MQRC_SIGNAL_REQUEST_ACCEPTED	2070	X'00000816'
MQRC_STORAGE_NOT_AVAILABLE	2071	X'00000817'
MQRC_SYNCPOINT_NOT_AVAILABLE	2072	X'00000818'
MQRC_TRIGGER_CONTROL_ERROR	2075	X'0000081B'
MQRC_TRIGGER_DEPTH_ERREUR	2076	X'0000081C'
MQRC_TRIGGER_MSG_PRIORITY_ERR	2077	X'0000081D'
MQRC_TRIGGER_TYPE_ERREUR	2078	X'0000081E'
MQRC_TRUNCATED_MSG_ACCEPTED	2079	X'0000081F'
MQRC_TRUNCATED_MSG_FAILED	2080	X'00000820'
MQRC_UNKNOWN_ALIAS_BASE_Q	2082	X'00000822'
MQRC_UNKNOWN_NOM_OBJET	2085	X'00000825'
MQRC_UNKNOWN_OBJET_Q_MGR	2086	X'00000826'
MQRC_UNKNOWN_REMOTE_Q_MGR	2087	X'00000827'
MQRC_WAIT_INTERVAL_ERROR	2090	X'0000082A'
MQRC_XMIT_Q_TYPE_ERROR	2091	X'0000082B'
MQRC_XMIT_Q_USAGE_ERROR	2092	X'0000082C'
MQRC_NOT_OPEN_FOR_PASS_ALL	2093	X'0000082D'
MQRC_NOT_OPEN_FOR_PASS_IDENT	2094	X'0000082E'
MQRC_NOT_OPEN_FOR_SET_ALL	2095	X'0000082F'
MQRC_NOT_OPEN_FOR_SET_IDENT	2096	X'00000830'
MQRC_CONTEXT_HANDLE_ERREUR	2097	X'00000831'
MQRC_CONTEXT_NOT_AVAILABLE	2098	X'00000832'
MQRC_SIGNAL1_ERROR	2099	X'00000833'
MQRC_OBJECT_ALREADY_EXISTS	2100	X'00000834'
MQRC_OBJECT_DAMAGED	2101	X'00000835'
MQRC_RESOURCE_PROBLEM	2102	X'00000836'
MQRC_ANOTHER_Q_MGR_CONNECTED	2103	X'00000837'
MQRC_UNKNOWN_REPORT_OPTION	2104	X'00000838'
MQRC_STORAGE_CLASS_ERROR	2105	X'00000839'
MQRC_COD_NOT_VALID_FOR_XCF_Q	2106	X'0000083A'
MQRC_XWAIT_ANNULE	2107	X'0000083B'
MQRC_XWAIT_ERREUR	2108	X'0000083C'
MQRC_SUPPRESSED_BY_EXIT	2109	X'0000083D'
MQRC_FORMAT_ERROR	2110	X'0000083E'

Tableau 304. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_SOURCE_CCSDID_ERROR	2111	X'0000083F'
MQRC_SOURCE_INTEGER_ENC_ERROR	2112	X'00000840'
MQRC_SOURCE_DECIMAL_ENC_ERROR	2113	X'00000841'
MQRC_SOURCE_FLOAT_ENC_ERROR	2114	X'00000842'
MQRC_TARGET_CCSDID_ERROR	2115	X'00000843'
MQRC_TARGET_INTEGER_ENC_ERROR	2116	X'00000844'
MQRC_TARGET_DECIMAL_ENC_ERROR	2117	X'00000845'
MQRC_TARGET_FLOAT_ENC_ERROR	2118	X'00000846'
MQRC_NOT_CONVERTED	2119	X'00000847'
MQRC_CONVERTED_MSG_TOO_BIG	2120	X'00000848'
MQRC_TRONQUÉ	2120	X'00000848'
MQRC_NO_EXTERNAL_PARTICIPANTS	2121	X'00000849'
MQRC_PARTICIPANT_NOT_XX_ENCODE_CASE_ONE disponible	2122	X'0000084A'
MQRC_OUTCOME_MIXTE	2123	X'0000084B'
MQRC_OUTCOME_EN attente	2124	X'0000084C'
MQRC_BRIDGE_STARTED	2125	X'0000084D'
MQRC_BRIDGE_STOPPED	2126	X'0000084E'
MQRC_ADAPTER_STOCKAGE_RUPTURE	2127	X'0000084F'
MQRC_UOW_EN COURS	2128	X'00000850'
MQRC_ADAPTER_CONN_LOAD_ERROR	2129	X'00000851'
MQRC_ADAPTER_SERV_LOAD_ERROR	2130	X'00000852'
MQRC_ADAPTER_DEFS_ERREUR	2131	X'00000853'
MQRC_ADAPTER_DEFS_LOAD_ERROR	2132	X'00000854'
MQRC_ADAPTER_CONV_LOAD_ERROR	2133	X'00000855'
MQRC_BO_ERREUR	2134	X'00000856'
MQRC_DH_ERREUR	2135	X'00000857'
MQRC_MULTIPLE_MOTIFS	2136	X'00000858'
MQRC_OPEN_FAILED	2137	X'00000859'
MQRC_ADAPTER_DISC_LOAD_ERROR	2138	X'0000085A'
MQRC_CNO_ERREUR	2139	X'0000085B'
MQRC_CICS_WAIT_FAILED	2140	X'0000085C'
MQRC_DLH_ERREUR	2141	X'0000085D'
MQRC_HEADER_ERREUR	2142	X'0000085E'
MQRC_SOURCE_LENGTH_ERROR	2143	X'0000085F'
MQRC_TARGET_LENGTH_ERROR	2144	X'00000860'
MQRC_SOURCE_BUFFER_ERROR	2145	X'00000861'
MQRC_TARGET_BUFFER_ERROR	2146	X'00000862'
MQRC_IIH_ERREUR	2148	X'00000864'
MQRC_PCF_ERREUR	2149	X'00000865'

Tableau 304. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_DBCS_ERROR	2150	X'00000866'
MQRC_OBJECT_NAME_ERROR	2152	X'00000868'
MQRC_OBJECT_Q_MGR_NAME_ERROR	2153	X'00000869'
MQRC_RECS_Présentateur-ERREUR	2154	X'0000086A'
MQRC_OBJECT_RECORDS_ERROR	2155	X'0000086B'
MQRC_RESPONSE_RECORDS_ERROR	2156	X'0000086C'
MQRC_ASID_MISMATCH	2157	X'0000086D'
MQRC_PMO_RECORD_FLAGS_ERROR	2158	X'0000086E'
MQRC_PUT_MSG_RECORDS_ERROR	2159	X'0000086F'
MQRC_CONN_ID_IN_USE	2160	X'00000870'
MQRC_Q_MGR QUIESCING	2161	X'00000871'
MQRC_Q_MGR_STOPPING	2162	X'00000872'
MQRC_DUPLICATE_RECOV_COORD	2163	X'00000873'
MQRC_PMO_ERREUR	2173	X'0000087D'
MQRC_API_EXIT_NOT_FOUND	2182	X'00000886'
MQRC_API_EXIT_LOAD_ERROR	2183	X'00000887'
MQRC_REMOTE_Q_NAME_ERROR	2184	X'00000888'
MQRC_INCONSISTENT_PERSISTENCE	2185	X'00000889'
MQRC_GMO_ERROR	2186	X'0000088A'
MQRC_CICS_BRIDGE_RESTRICTION	2187	X'0000088B'
MQRC_STOPPED_BY_CLUSTER_EXIT	2188	X'0000088C'
Erreur de résolution MQRC_CLUSTER_RESOLUTION_ERROR	2189	X'0000088D'
MQRC_CONVERTED_STRING_TOO_BIG	2190	X'0000088E'
MQRC_TMC_ERREUR	2191	X'0000088F'
MQRC_PAGESET_FULL	2192	X'00000890'
MQRC_STORAGE_MEDIUM_FULL	2192	X'00000890'
MQRC_PAGESET_ERROR	2193	X'00000891'
MQRC_NAME_NOT_VALID_FOR_TYPE	2194	X'00000892'
MQRC_UNEXPECTED_ERROR	2195	X'00000893'
MQRC_UNKNOWN_XMIT_Q	2196	X'00000894'
MQRC_UNKNOWN_DEF_XMIT_Q	2197	X'00000895'
MQRC_DEF_XMIT_Q_TYPE_ERREUR	2198	X'00000896'
MQRC_DEF_XMIT_Q_USAGE_ERROR	2199	X'00000897'
MQRC_MSG_MARKED_BROWSE_CO_OP	2200	X'00000898'
MQRC_NAME_IN_USE	2201	X'00000899'
MQRC_CONNECTION QUIESCING	2202	X'0000089A'
MQRC_CONNECTION_STOPPING	2203	X'0000089B'
MQRC_ADAPTER_NOT_AVAILABLE	2204	X'0000089C'
MQRC_MSG_ID_ERROR	2206	X'0000089E'

Tableau 304. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_CORREL_ID_ERROR	2207	X'0000089F'
MQRC_FILE_SYSTEM_ERROR	2208	X'000008A0'
MQRC_NO_MSG_LOCKED	2209	X'000008A1'
MQRC_SOAP_DOTNET_ERROR	2210	X'000008A2'
ERREUR MQRC_SOAP_AXIS_ERROR	2211	X'000008A3'
MQRC_SOAP_URL_ERROR	2212	X'000008A4'
MQRC_FILE_NON_AUDITÉE	2216	X'000008A8'
MQRC_CONNECTION_NOT_AUTHORIZED	2217	X'000008A9'
MQRC_MSG_TOO_BIG_FOR_CHANNEL	2218	X'000008AA'
MQRC_CALL_IN_PROGRESS	2219	X'000008AB'
MQRC_RMH_ERREUR	2220	X'000008AC'
MQRC_Q_MGR_ACTIVIF	2222	X'000008AE'
MQRC_Q_MGR_NOT_ACTIVE	2223	X'000008AF'
MQRC_Q_DEPTH_ELEVE	2224	X'000008B0'
MQRC_Q_DEPTH_LOW	2225	X'000008B1'
MQRC_Q_SERVICE_INTERVAL_HIGH	2226	X'000008B2'
MQRC_Q_SERVICE_INTERVAL_OK	2227	X'000008B3'
MQRC_RFH_HEADER_FIELD_ERROR	2228	X'000008B4'
MQRC_RAS_PROPERTY_ERROR	2229	X'000008B5'
MQRC_UNIT_OF_WORK_NOT_STARTED	2232	X'000008B8'
MQRC_CHANNEL_AUTO_DEF_OK	2233	X'000008B9'
MQRC_CHANNEL_AUTO_ERREUR PAR DEFAUT	2234	X'000008BA'
MQRC_CFH_ERREUR	2235	X'000008BB'
MQRC_CFIL_ERREUR	2236	X'000008BC'
MQRC_CFIN_ERREUR	2237	X'000008BD'
MQRC_CFSL_ERREUR	2238	X'000008BE'
MQRC_CFST_ERREUR	2239	X'000008BF'
MQRC_INCOMPLETE_GROUP	2241	X'000008C1'
MQRC_INCOMPLETE_MSG	2242	X'000008C2'
MQRC_INCONSISTENT_CCSDS	2243	X'000008C3'
MQRC_INCONSISTENT_ENCODINGS	2244	X'000008C4'
MQRC_INCONSISTENT_UOW	2245	X'000008C5'
MQRC_INVALID_MSG_UNDER_CURSOR	2246	X'000008C6'
MQRC_MATCH_OPTIONS_ERROR	2247	X'000008C7'
MQRC_MDE_ERREUR	2248	X'000008C8'
MQRC_MSG_FLAGS_ERREUR	2249	X'000008C9'
MQRC_MSG_SEQ_NUMBER_ERROR	2250	X'000008CA'
ERREUR MQRC_OFFSET_ERROR	2251	X'000008CB'
MQRC_ORIGINAL_LENGTH_ERROR	2252	X'000008CC'

Tableau 304. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQRC_SEGMENT_LONGUEUR_ZÉRO	2253	X'000008CD'
MQRC_UOW_NOT_AVAILABLE	2255	X'000008CF'
MQRC_WRONG_GMO_VERSION	2256	X'000008D0'
MQRC_WRONG_MD_VERSION	2257	X'000008D1'
MQRC_GROUP_ID_ERREUR	2258	X'000008D2'
MQRC_INCONSISTENT_BROWSE	2259	X'000008D3'
MQRC_XQH_ERREUR	2260	X'000008D4'
MQRC_SRC_ENV_ERREUR	2261	X'000008D5'
MQRC_SRC_NAME_ERROR	2262	X'000008D6'
MQRC_DEST_ENV_ERREUR	2263	X'000008D7'
MQRC_DEST_NAME_ERREUR	2264	X'000008D8'
MQRC_TM_ERREUR	2265	X'000008D9'
MQRC_CLUSTER_EXIT_ERREUR	2266	X'000008DA'
MQRC_CLUSTER_EXIT_LOAD_ERROR	2267	X'000008DB'
MQRC_CLUSTER_PUT_INHIBÉ	2268	X'000008DC'
MQRC_CLUSTER_RESOURCE_ERROR	2269	X'000008DD'
MQRC_NO_DESTINATIONS_AVAILABLE	2270	X'000008DE'
MQRC_CONN_TAG_IN_USE	2271	X'000008DF'
MQRC_PARTIALLY_CONVERTED	2272	X'000008E0'
MQRC_CONNECTION_ERREUR	2273	X'000008E1'
MQRC_OPTION_ENVIRONMENT_ERROR	2274	X'000008E2'
MQRC_CD_ERREUR	2277	X'000008E5'
MQRC_CLIENT_CONN_ERROR	2278	X'000008E6'
MQRC_CHANNEL_STOPPED_BY_USER	2279	X'000008E7'
ERREUR MQRC_HCONFIG_ERROR	2280	X'000008E8'
Erreur de fonction MQRC_FUNCTION_ERROR	2281	X'000008E9'
MQRC_CHANNEL_STARTED	2282	X'000008EA'
MQRC_CHANNEL_STOPPED	2283	X'000008EB'
MQRC_CHANNEL_CONV_ERROR	2284	X'000008EC'
MQRC_SERVICE_NOT_AVAILABLE	2285	X'000008ED'
MQRC_INITIALIZATION_FAILED	2286	X'000008EE'
Echec de la commande MQRC_TERMINATION_FAILED	2287	X'000008EF'
MQRC_UNKNOWN_NOM_QQ	2288	X'000008F0'
MQRC_SERVICE_ERROR	2289	X'000008F1'
MQRC_Q_ALREADY_EXISTS	2290	X'000008F2'
MQRC_USER_ID_NOT_AVAILABLE	2291	X'000008F3'
MQRC_UNKNOWN_ENTITY	2292	X'000008F4'
MQRC_UNKNOWN_ENTITÉ_AUTH	2293	X'000008F5'
MQRC_UNKNOWN_REF_OBJECT	2294	X'000008F6'

Tableau 304. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_CHANNEL_ACTIVATED	2295	X'000008F7'
MQRC_CHANNEL_NOT_ACTIVATED	2296	X'000008F8'
MQRC_UOW_ANNULE	2297	X'000008F9'
MQRC_FUNCTION_NOT_SUPPORTED	2298	X'000008FA'
MQRC_SELECTOR_TYPE_ERREUR	2299	X'000008FB'
MQRC_COMMAND_TYPE_ERROR	2300	X'000008FC'
MQRC_MULTIPLE_INSTANCE_ERROR	2301	X'000008FD'
MQRC_SYSTEM_ITEM_NOT_ALTERABLE	2302	X'000008FE'
MQRC_BAG_CONVERSION_ERREUR	2303	X'000008FF'
MQRC_SELECTOR_OUT_OF_RANGE	2304	X'00000900'
MQRC_SELECTOR_NOT_UNIQUE	2305	X'00000901'
MQRC_INDEX_NOT_PRESENT	2306	X'00000902'
ERREUR MQRC_STRING_ERROR	2307	X'00000903'
MQRC_ENCODING_NOT_SUPPORTED	2308	X'00000904'
MQRC_SELECTOR_NOT_PRESENT	2309	X'00000905'
MQRC_OUT_SELECTOR_ERREUR	2310	X'00000906'
MQRC_STRING_TRONQUÉ	2311	X'00000907'
MQRC_SELECTOR_WRONG_TYPE	2312	X'00000908'
MQRC_INCONSISTENT_ITEM_TYPE	2313	X'00000909'
ERREUR MQRC_INDEX_ERREUR	2314	X'0000090A'
MQRC_SYSTEM_BAG_NOT_ALTERABLE	2315	X'0000090B'
MQRC_ITEM_COUNT_ERREUR	2316	X'0000090C'
MQRC_FORMAT_NOT_SUPPORTED	2317	X'0000090D'
MQRC_SELECTOR_NOT_SUPPORTED	2318	X'0000090E'
MQRC_ITEM_VALEUR_ERREUR	2319	X'0000090F'
MQRC_HBAG_ERREUR	2320	X'00000910'
MQRC_PARAMETER_MISSING	2321	X'00000911'
MQRC_CMD_SERVER_NOT_AVAILABLE	2322	X'00000912'
MQRC_STRING_LENGTH_ERROR	2323	X'00000913'
MQRC_INQUIRY_COMMAND_ERROR	2324	X'00000914'
MQRC_NESTED_BAG_NOT_SUPPORTED	2325	X'00000915'
TYPE MQRC_BAG_WRONG_TYPE	2326	X'00000916'
MQRC_ITEM_TYPE_ERROR	2327	X'00000917'
MQRC_SYSTEM_BAG_NOT_DELETABLE	2328	X'00000918'
MQRC_SYSTEM_ITEM_NOT_DELETABLE	2329	X'00000919'
MQRC_CODED_CHAR_SET_ID_ERROR	2330	X'0000091A'
MQRC_MSG_TOKEN_ERROR	2331	X'0000091B'
MQRC_WIH	2332	X'0000091C'
MQRC_WIH_ERREUR	2333	X'0000091D'

Tableau 304. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQRC_RFH_ERREUR	2334	X'0000091E'
MQRC_RFH_STRING_ERROR	2335	X'0000091F'
MQRC_RFH_ERREUR DE COMMANDE	2336	X'00000920'
MQRC_RFH_PARM_ERREUR	2337	X'00000921'
MQRC_RFH_DUPLICATE_PARM	2338	X'00000922'
MQRC_RFH_PARM_MISSING	2339	X'00000923'
MQRC_CHAR_CONVERSION_ERREUR	2340	X'00000924'
MQRC_UCS2_CONVERSION_ERROR	2341	X'00000925'
MQRC_DB2_NOT_AVAILABLE	2342	X'00000926'
MQRC_OBJECT_NOT_UNIQUE	2343	X'00000927'
MQRC_CONN_TAG_NOT_RELÂCHÉE	2344	X'00000928'
MQRC_CF_NOT_XX_ENCODE_CASE_ONE disponible	2345	X'00000929'
MQRC_CF_STRUC_IN_USE	2346	X'0000092A'
MQRC_CF_STRUC_LIST_HDR_IN_USE	2347	X'0000092B'
Echec de MQRC_CF_STRUC_AUTH_FAILED	2348	X'0000092C'
MQRC_CF_STRUC_ERREUR	2349	X'0000092D'
MQRC_CONN_TAG_NON_UTILISABLE	2350	X'0000092E'
MQRC_GLOBAL_UOW_CONFLICT	2351	X'0000092F'
MQRC_LOCAL_UOW_CONFLICT	2352	X'00000930'
MQRC_HANDLE_IN_USE_FOR_UOW	2353	X'00000931'
MQRC_UOW_ENLISTMENT_ERROR	2354	X'00000932'
MQRC_UOW_MIX_NOT_SUPPORTED	2355	X'00000933'
MQRC_WXP_ERREUR	2356	X'00000934'
MQRC_CURRENT_RECORD_ERROR	2357	X'00000935'
MQRC_NEXT_OFFSET_ERREUR	2358	X'00000936'
MQRC_NO_RECORD_DISPONIBLE	2359	X'00000937'
MQRC_OBJET_NIVEAU_INCOMPATIBLE	2360	X'00000938'
MQRC_NEXT_RECORD_ERREUR	2361	X'00000939'
MQRC_BACKOUT_SEUIL_ATTEINT	2362	X'0000093A'
MQRC_MSG_NON_CORRESPONDANCE	2363	X'0000093B'
MQRC_JMS_FORMAT_ERREUR	2364	X'0000093C'
MQRC_SEGMENTS_NON_PRIS en charge	2365	X'0000093D'
MQRC_WRONG_CF_NIVEAU	2366	X'0000093E'
MQRC_CONFIG_CREATE_OBJECT	2367	X'0000093F'
MQRC_CONFIG_CHANGE_OBJECT	2368	X'00000940'
MQRC_CONFIG_DELETE_OBJECT	2369	X'00000941'
MQRC_CONFIG_REFRESH_OBJECT	2370	X'00000942'
MQRC_CHANNEL_SSL_ERROR	2371	X'00000943'
MQRC_PARTICIPANT_NOT_DEFINI	2372	X'00000944'

Tableau 304. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_CF_STRUC_FAILED	2373	X'00000945'
MQRC_API_EXIT_ERROR	2374	X'00000946'
MQRC_API_EXIT_INIT_ERROR	2375	X'00000947'
MQRC_API_EXIT_TERM_ERREUR	2376	X'00000948'
MQRC_EXIT_MOTIF-ERREUR	2377	X'00000949'
MQRC_RESERVED_VALEUR_ERREUR	2378	X'0000094A'
MQRC_NO_DATA_XX_ENCODE_CASE_ONE disponible	2379	X'0000094B'
MQRC_ERREUR DE SCO_DU	2380	X'0000094C'
MQRC_KEY_REPOSITORY_ERROR	2381	X'0000094D'
MQRC_CRYPTO_ERREUR_MATÉRIEL	2382	X'0000094E'
MQRC_AUTH_INFO_REC_COUNT_ERROR	2383	X'0000094F'
MQRC_AUTH_INFO_REC_ERROR	2384	X'00000950'
ERREUR MQRC_AIR_ERROR	2385	X'00000951'
MQRC_AUTH_INFO_TYPE_ERROR	2386	X'00000952'
MQRC_AUTH_INFO_CONN_NAME_ERROR	2387	X'00000953'
MQRC_LDAP_USER_NAME_ERROR	2388	X'00000954'
MQRC_LDAP_NOM_UTILISATEUR_LENGTH_ERR	2389	X'00000955'
MQRC_LDAP_PASSWORD_ERROR	2390	X'00000956'
MQRC_SSL_ALREADY_INITIALISÉ	2391	X'00000957'
MQRC_SSL_CONFIG_ERROR	2392	X'00000958'
MQRC_SSL_INITIALIZATION_ERROR	2393	X'00000959'
MQRC_Q_INDEX_TYPE_ERROR	2394	X'0000095A'
MQRC_CFBS_ERREUR	2395	X'0000095B'
MQRC_SSL_NOT_ALLOWED	2396	X'0000095C'
MQRC_JSSE_ERREUR	2397	X'0000095D'
MQRC_SSL_PEER_NAME_MISMATCH	2398	X'0000095E'
MQRC_SSL_PEER_NAME_ERROR	2399	X'0000095F'
MQRC_UNSUPPORTED_CIPHER_SUITE	2400	X'00000960'
MQRC_SSL_CERTIFICATE RÉVOQUÉ	2401	X'00000961'
MQRC_SSL_CERT_STORE_ERREUR	2402	X'00000962'
MQRC_CLIENT_EXIT_LOAD_ERROR	2406	X'00000966'
MQRC_CLIENT_EXIT_ERREUR	2407	X'00000967'
MQRC_UOW_COMMITTED	2408	X'00000968'
MQRC_SSL_KEY_RESET_ERROR	2409	X'00000969'
MQRC_UNKNOWN_NOM_COMPOSANT	2410	X'0000096A'
STATUT DU GESTIONNAIRE DE FILES D'ATTENTE	2411	X'0000096B'
MQRC_COMMAND_MQSC	2412	X'0000096C'
MQRC_COMMAND_PCF	2413	X'0000096D'
MQRC_CFIF_ERREUR	2414	X'0000096E'



Tableau 304. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_CFSF_ERREUR	2415	X'0000096F'
MQRC_CFGR_ERREUR	2416	X'00000970'
MQRC_MSG_NOT_ALLOWED_IN_GROUP	2417	X'00000971'
MQRC_FILTER_OPERATOR_ERROR	2418	X'00000972'
MQRC_NESTED_SELECTOR_ERREUR	2419	X'00000973'
MQRC_EPH_ERREUR	2420	X'00000974'
MQRC_RFH_FORMAT_ERREUR	2421	X'00000975'
MQRC_CFBF_ERREUR	2422	X'00000976'
MQRC_CLIENT_CHANNEL_CONFLICT	2423	X'00000977'
MQRC_SD_ERREUR	2424	X'00000978'
MQRC_TOPIC_STRING_ERROR	2425	X'00000979'
MQRC_STS_ERREUR	2426	X'0000097A'
MQRC_NO_ABONNEMENT	2428	X'0000097C'
MQRC_SUBSCRIPTION_IN_USE	2429	X'0000097D'
MQRC_STAT_TYPE_ERREUR	2430	X'0000097E'
MQRC_SUB_USER_DATA_ERROR	2431	X'0000097F'
MQRC_SUB_ALREADY_EXISTS	2432	X'00000980'
MQRC_IDENTITY_MISMATCH	2434	X'00000982'
MQRC_ALTER_SUB_ERREUR	2435	X'00000983'
MQRC_DURABILITY_NOT_ALLOWED	2436	X'00000984'
MQRC_NO_RETAINED_MSG	2437	X'00000985'
MQRC_SRO_ERREUR	2438	X'00000986'
MQRC_SUB_NAME_ERREUR	2440	X'00000988'
MQRC_OBJECT_STRING_ERROR	2441	X'00000989'
MQRC_PROPERTY_NAME_ERROR	2442	X'0000098A'
MQRC_SEGMENTATION_NOT_ALLOWED	2443	X'0000098B'
MQRC_CBD_ERREUR	2444	X'0000098C'
MQRC_CTLO_ERREUR	2445	X'0000098D'
MQRC_NO_CALLBACKS_ACTIVE	2446	X'0000098E'
MQRC_CALLBACK_NOT_REGISTERED	2448	X'00000990'
MQRC_OPTIONS_CHANGED	2457	X'00000999'
MQRC_READ_AHEAD_MSGS	2458	X'0000099A'
MQRC_SELECTOR_SYNTAX_ERREUR	2459	X'0000099B'
ERREUR MQRC_HMSG_ERROR	2460	X'0000099C'
ERREUR MQRC_CMHO_ERROR	2461	X'0000099D'
MQRC_DMHO_ERREUR	2462	X'0000099E'
MQRC_SMPO_ERREUR	2463	X'0000099F'
MQRC_IMPO_ERREUR	2464	X'000009A0'
MQRC_PROPERTY_NAME_TOO_BIG	2465	X'000009A1'

Tableau 304. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_PROP_VALEUR_NOT_CONVERTED	2466	X'000009A2'
MQRC_PROP_TYPE_NON_PRIS en charge	2467	X'000009A3'
MQRC_PROPERTY_VALEUR_TOO_BIG	2469	X'000009A5'
MQRC_PROP_CONV_NOT_SUPPORTED	2470	X'000009A6'
MQRC_PROPERTY_NOT_AVAILABLE	2471	X'000009A7'
MQRC_PROP_NUMBER_FORMAT_ERROR	2472	X'000009A8'
MQRC_PROPERTY_TYPE_ERROR	2473	X'000009A9'
MQRC_PROPERTIES_TOO_BIG	2478	X'000009AE'
MQRC_PUT_NOT_RETENU	2479	X'000009AF'
MQRC_ALIAS_TARGTYPE_CHANGED	2480	X'000009B0'
MQRC_DMPO_ERREUR	2481	X'000009B1'
MQRC_PD_ERREUR	2482	X'000009B2'
MQRC_CALLBACK_TYPE_ERREUR	2483	X'000009B3'
MQRC_CBD_OPTIONS_ERREUR	2484	X'000009B4'
MQRC_MAX_MSG_LENGTH_ERROR	2485	X'000009B5'
MQRC_CALLBACK_ROUTINE_ERROR	2486	X'000009B6'
MQRC_CALLBACK_LINK_ERROR	2487	X'000009B7'
ERREUR D'OPERATION_MQRC	2488	X'000009B8'
MQRC_BMHO_ERREUR	2489	X'000009B9'
MQRC_UNSUPPORTED_PROPERTY	2490	X'000009BA'
MQRC_PROP_NOM_NON_CONVERTI	2492	X'000009BC'
MQRC_GET_ENABLED	2494	X'000009BE'
MQRC_MODULE_NOT_FOUND	2495	X'000009BF'
MQRC_MODULE_INVALID	2496	X'000009C0'
MQRC_MODULE_ENTRY_NOT_FOUND	2497	X'000009C1'
MQRC_MIXED_CONTENT_NOT_ALLOWED	2498	X'000009C2'
MQRC_MSG_HANDLE_IN_USE	2499	X'000009C3'
MQRC_HCONN_ASYNC_ACTIVE	2500	X'000009C4'
MQRC_MHBO_ERREUR	2501	X'000009C5'
MQRC_PUBLICATION_FAILURE	2502	X'000009C6'
MQRC_SUB_INHIBÉ	2503	X'000009C7'
MQRC_SELECTOR_ALWAYS_FALSE	2504	X'000009C8'
MQRC_XEPO_ERREUR	2507	X'000009CB'
MQRC_DURATIONITY_NOT_ALTERABLE	2509	X'000009CD'
MQRC_TOPIC_NOT_ALTERABLE	2510	X'000009CE'
MQRC_SUBLEVEL_NOT_ALTERABLE	2512	X'000009D0'
MQRC_PROPERTY_NAME_LENGTH_ERR	2513	X'000009D1'
MQRC_DUPLICATE_GROUPE_SUB	2514	X'000009D2'
MQRC_GROUPING_NOT_ALTERABLE	2515	X'000009D3'

Tableau 304. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_SELECTOR_INVALID_FOR_TYPE	2516	X'000009D4'
MQRC_HOBJ_MISE au repos	2517	X'000009D5'
MQRC_HOBJ_MSGS mis au repos	2518	X'000009D6'
MQRC_SELECTION_STRING_ERROR	2519	X'000009D7'
MQRC_RES_OBJECT_STRING_ERROR	2520	X'000009D8'
MQRC_CONNECTION_SUSPENDUE	2521	X'000009D9'
MQRC_INVALID_DESTINATION	2522	X'000009DA'
MQRC_INVALID_ABONNEMENT	2523	X'000009DB'
MQRC_SELECTOR_NOT_ALTERABLE	2524	X'000009DC'
MQRC_RETAINED_MSG_Q_ERREUR	2525	X'000009DD'
MQRC_RETAINED_NOT_LIVRÉ	2526	X'000009DE'
MQRC_RFH_RESTRICTED_FORMAT_ERR	2527	X'000009DF'
MQRC_CONNECTION_STOPPED	2528	X'000009E0'
MQRC_ASYNC_UOW_CONFLICT	2529	X'000009E1'
MQRC_ASYNC_XA_CONFLICT	2530	X'000009E2'
MQRC_PUBSUB_INHIBÉ	2531	X'000009E3'
MQRC_MSG_HANDLE_COPY_FAILURE	2532	X'000009E4'
MQRC_DEST_CLASS_NOT_ALTERABLE	2533	X'000009E5'
MQRC_OPERATION_NOT_ALLOWED	2534	X'000009E6'
ERREUR MQRC_ACTION_ERROR	2535	X'000009E7'
MQRC_CHANNEL_NOT_AVAILABLE	2537	X'000009E9'
MQRC_HOST_NOT_AVAILABLE	2538	X'000009EA'
MQRC_CHANNEL_CONFIG_ERROR	2539	X'000009EB'
MQRC_UNKNOWN_NOM_CANAL	2540	X'000009EC'
MQRC_LOOPING_PUBLICATION	2541	X'000009ED'
MQRC_ALREADY_JOINT	2542	X'000009EE'
MQRC_CHANNEL_SSL_WARNING	2552	X'000009F8'
MQRC_OCSP_URL_ERROR	2553	X'000009F9'
MQRC_CIPHER_SPEC_NOT_SUITE_B	2591	X'00000A1F'
MQRC_SUITE_B_ERREUR	2592	X'00000A20'
MQRC_PASSWORD_PROTECTION_ERROR	2594	X'00000A22'
MQRC_REOPEN_EXCL_INPUT_ERROR	6100	X'000017D4'
MQRC_REOPEN_INQUIRE_ERROR	6101	X'000017D5'
MQRC_REOPEN_SAVED_CONTEXT_ERR	6102	X'000017D6'
MQRC_REOPEN_TEMPORARY_Q_ERROR	6103	X'000017D7'
MQRC_ATTRIBUT_LOCKED	6104	X'000017D8'
MQRC_CURSOR_NOT_VALID	6105	X'000017D9'
ERREUR_CODAGE_MQRC_ERREUR	6106	X'000017DA'
MQRC_STRUC_ID_ERROR	6107	X'000017DB'

Tableau 304. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
POINT_NULL_MQRC_NULL	6108	X'000017DC'
MQRC_NO RÉFÉRENCE_CONNEXION	6109	X'000017DD'
MQRC_NO_BUFFER	6110	X'000017DE'
MQRC_BINARY_DATA_LENGTH_ERROR	6111	X'000017DF'
MQRC_BUFFER_NOT_AUTOMATIC	6112	X'000017E0'
MQRC_INSUFFISANT_BUFFER	6113	X'000017E1'
MQRC_DONNÉES_INSUFFISANTES	6114	X'000017E2'
MQRC_DATA_TRONQUÉ	6115	X'000017E3'
LONGUEUR_ZÉRO_MQRC_	6116	X'000017E4'
LONGUEUR_NÉGATIVE_MQRC	6117	X'000017E5'
DÉCALAGE_NÉGATIVE_MQRC	6118	X'000017E6'
MQRC_INCONSISTENT_FORMAT	6119	X'000017E7'
MQRC_INCONSISTENT_OBJECT_STATE	6120	X'000017E8'
MQRC_CONTEXT_OBJET_NON_VALIDE	6121	X'000017E9'
MQRC_CONTEXT_OPEN_ERROR	6122	X'000017EA'
MQRC_STRUC_LENGTH_ERROR	6123	X'000017EB'
MQRC_NOT_CONNECTED	6124	X'000017EC'
MQRC_NOT_OPEN	6125	X'000017ED'
MQRC_DISTRIBUTION_LIST_EMPTY	6126	X'000017EE'
MQRC_INCONSISTENT_OPEN_OPTIONS	6127	X'000017EF'
VERSION MQRC_WRONG_VERSION	6128	X'000017F0'
ERREUR MQRC_REFERENCE_ERROR	6129	X'000017F1'

## MQRCCF\_\* (codes anomalie d'en-tête de format de commande)

Pour plus d'informations sur la réponse du programmeur, voir [Codes raison PCF](#).

Tableau 305. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQRCCF_CFH_TYPE_ERROR	3001	X'00000BB9'
MQRCCF_CFH_LENGTH_ERROR	3002	X'00000BBA'
MQRCCF_CFH_VERSION_ERROR	3003	X'00000BBB'
MQRCCF_CFH_MSG_SEQ_NUMBER_ERR	3004	X'00000BBC'
MQRCCF_CFH_CONTROL_ERROR	3005	X'00000BBD'
MQRCCF_CFH_PARM_COUNT_ERROR	3006	X'00000BBE'
MQRCCF_CFH_COMMAND_ERROR	3007	X'00000BBF'
Echec de la commande MQRCCF_COMMAND_FAILED	3008	X'00000BC0'
MQRCCF_CFIN_LENGTH_ERROR	3009	X'00000BC1'
MQRCCF_CFST_LENGTH_ERROR	3010	X'00000BC2'
MQRCCF_CFST_STRING_LENGTH_ERR	3011	X'00000BC3'
MQRCCF_FORCE_VALEUR_ERREUR	3012	X'00000BC4'

Tableau 305. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRCCF_STRUCTURE_TYPE_ERROR	3013	X'00000BC5'
MQRCCF_CFIN_PARM_ID_ERROR	3014	X'00000BC6'
MQRCCF_CFST_PARM_ID_ERROR	3015	X'00000BC7'
MQRCCF_MSG_LENGTH_ERROR	3016	X'00000BC8'
MQRCCF_CFIN_DUPLICATE_PARM	3017	X'00000BC9'
MQRCCF_CFST_DUPLICATE_PARM	3018	X'00000BCA'
MQRCCF_PARM_COUNT_TOO_SMALL	3019	X'00000BCB'
MQRCCF_COMPTE_PARM_TOO_BIG	3020	X'00000BCC'
MQRCCF_Q_ALREADY_IN_CELL	3021	X'00000BCD'
MQRCCF_Q_TYPE_ERROR	3022	X'00000BCE'
MQRCCF_MD_FORMAT_ERROR	3023	X'00000BCF'
MQRCCF_CFSL_LENGTH_ERROR	3024	X'00000BD0'
MQRCCF_REPLACE_VALEUR_ERREUR	3025	X'00000BD1'
MQRCCF_CFIL_DUPLICATE_VALUE	3026	X'00000BD2'
MQRCCF_CFIL_COUNT_ERROR	3027	X'00000BD3'
MQRCCF_CFIL_LENGTH_ERROR	3028	X'00000BD4'
MQRCCF_QUIESCE_VALEUR_ERREUR	3029	X'00000BD5'
MQRCCF_MODE_VALEUR_ERREUR	3029	X'00000BD5'
MQRCCF_MSG_SEQ_NUMBER_ERROR	3030	X'00000BD6'
MQRCCF_PING_DATA_COUNT_ERROR	3031	X'00000BD7'
MQRCCF_PING_DATA_COMPARE_ERROR	3032	X'00000BD8'
MQRCCF_CFSL_PARM_ID_ERROR	3033	X'00000BD9'
MQRCCF_CHANNEL_TYPE_ERROR	3034	X'00000BDA'
MQRCCF_PARM_SEQUENCE_ERROR	3035	X'00000BDB'
MQRCCF_XMIT_PROTOCOL_TYPE_ERR	3036	X'00000BDC'
MQRCCF_TAILLE_BATCH_ERREUR	3037	X'00000BDD'
ERREUR MQRCCF_DISC_INT_ERROR	3038	X'00000BDE'
MQRCCF_SHORT_RETRY_ERROR	3039	X'00000BDF'
MQRCCF_SHORT_TIMER_ERROR	3040	X'00000BE0'
MQRCCF_LONG_RETRY_ERROR	3041	X'00000BE1'
MQRCCF_LONG_TIMER_ERREUR	3042	X'00000BE2'
MQRCCF_SEQ_NUMBER_WRAP_ERROR	3043	X'00000BE3'
MQRCCF_MAX_MSG_LENGTH_ERROR	3044	X'00000BE4'
MQRCCF_PUT_AUTH_ERROR	3045	X'00000BE5'
MQRCCF_PURGE_VALEUR_ERREUR	3046	X'00000BE6'
MQRCCF_CFIL_PARM_ID_ERROR	3047	X'00000BE7'
MQRCCF_MSG_TRUNCATED	3048	X'00000BE8'
ERREUR MQRCCF_CCSID	3049	X'00000BE9'
MQRCCF_ERREUR_CODAGE	3050	X'00000BEA'

Tableau 305. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRCCF_QUEUES_VALUE_ERROR	3051	X'00000BEB'
MQRCCF_DATA_CONV_VALEUR_ERREUR	3052	X'00000BEC'
MQRCCF_INDOUBT_VALEUR_ERREUR	3053	X'00000BED'
MQRCCF_ESCAPE_TYPE_ERROR	3054	X'00000BEE'
MQRCCF_REPOS_VALEUR_ERREUR	3055	X'00000BEF'
MQRCCF_CHANNEL_TABLE_ERROR	3062	X'00000BF6'
MQRCCF_MCA_TYPE_ERROR	3063	X'00000BF7'
MQRCCF_CHL_INST_TYPE_ERROR	3064	X'00000BF8'
MQRCCF_CHL_STATUS_NOT_FOUND	3065	X'00000BF9'
MQRCCF_CFSL_DUPLICATE_PARM	3066	X'00000BFA'
MQRCCF_CFSL_TOTAL_LENGTH_ERROR	3067	X'00000BFB'
MQRCCF_CFSL_COUNT_ERROR	3068	X'00000BFC'
MQRCCF_CFSL_STRING_LENGTH_ERR	3069	X'00000BFD'
MQRCCF_BROKER_DELETED	3070	X'00000BFE'
MQRCCF_FLOT_ERREUR	3071	X'00000BFF'
MQRCCF_TOPIC_ERREUR	3072	X'00000C00'
MQRCCF_NOT_REGISTERED	3073	X'00000C01'
MQRCCF_Q_MGR_NAME_ERROR	3074	X'00000C02'
MQRCCF_INCORRECT_STREAM	3075	X'00000C03'
MQRCCF_Q_NOM_ERREUR	3076	X'00000C04'
MQRCCF_NO_RETAINED_MSG	3077	X'00000C05'
MQRCCF_IDENTITÉ_DOUBLON	3078	X'00000C06'
MQRCCF_INCORRECT_Q	3079	X'00000C07'
MQRCCF_CORREL_ID_ERREUR	3080	X'00000C08'
MQRCCF_NON_AUTORISÉ	3081	X'00000C09'
MQRCCF_FLUX_INCONNU	3082	X'00000C0A'
MQRCCF_REG_OPTIONS_ERREUR	3083	X'00000C0B'
MQRCCF_PUB_OPTIONS_ERROR	3084	X'00000C0C'
MQRCCF_COURTIER inconnu	3085	X'00000C0D'
MQRCCF_Q_MGR_CCSID_ERROR	3086	X'00000C0E'
MQRCCF_DEL_OPTIONS_ERROR	3087	X'00000C0F'
MQRCCF_CLUSTER_NAME_CONFLICT	3088	X'00000C10'
MQRCCF_NOM RÉFÉRENTIEL CONFLIT	3089	X'00000C11'
MQRCCF_CLUSTER_Q_USAGE_ERROR	3090	X'00000C12'
MQRCCF_ACTION_VALEUR_ERREUR	3091	X'00000C13'
MQRCCF_COMMS_LIBRARY_ERROR	3092	X'00000C14'
MQRCCF_NETBIOS_NAME_ERROR	3093	X'00000C15'
Echec de la commande MQRCCF_BROKER_COMMAND_FAILED	3094	X'00000C16'
MQRCCF_CFST_CONFLICTING_PARM	3095	X'00000C17'

Tableau 305. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRCCF_CHEMIN_NON_VALIDE	3096	X'00000C18'
MQRCCF_PARM_SYNTAX_ERROR	3097	X'00000C19'
MQRCCF_PWD_LENGTH_ERROR	3098	X'00000C1A'
MQRCCF_FILTER_ERREUR	3150	X'00000C4E'
MQRCCF_UTILISATEUR_FAUT_ERREUR	3151	X'00000C4F'
MQRCCF_ABONNEMENT_DOUBLON	3152	X'00000C50'
MQRCCF_SUB_NAME_ERREUR	3153	X'00000C51'
MQRCCF_SUB_IDENTITY_ERROR	3154	X'00000C52'
MQRCCF_SUBSCRIPTION_IN_USE	3155	X'00000C53'
MQRCCF_SUBSCRIPTION_LOCKED	3156	X'00000C54'
MQRCCF_ALREADY_JOINT	3157	X'00000C55'
MQRCCF_OBJET_EN_UTILISATION	3160	X'00000C58'
MQRCCF_NOM_FICHER_INCONNU	3161	X'00000C59'
MQRCCF_FILE_NON_DISPONIBLE	3162	X'00000C5A'
MQRCCF_DISC_RETRY_ERROR	3163	X'00000C5B'
MQRCCF_ALLOC_RETRY_ERROR	3164	X'00000C5C'
MQRCCF_ALLOC_SLOW_TIMER_ERROR	3165	X'00000C5D'
MQRCCF_ALLOC_FAST_TIMER_ERROR	3166	X'00000C5E'
MQRCCF_NUMÉRO_PORT_ERREUR	3167	X'00000C5F'
MQRCCF_CHL_SYSTEM_NOT_ACTIVE	3168	X'00000C60'
MQRCCF_ENTITY_NAME_MISSING	3169	X'00000C61'
MQRCCF_NOM_PROFIL_ERROR	3170	X'00000C62'
MQRCCF_AUTH_VALEUR_ERREUR	3171	X'00000C63'
MQRCCF_VALEUR_AUTH_MANQUANTE	3172	X'00000C64'
MQRCCF_TYPE_OBJET_MANQUANT	3173	X'00000C65'
MQRCCF_CONNECTION_ID_ERROR	3174	X'00000C66'
MQRCCF_LOG_TYPE_ERROR	3175	X'00000C67'
MQRCCF_PROGRAM_NON_DISPONIBLE	3176	X'00000C68'
MQRCCF_PROGRAM_AUTH_FAILED	3177	X'00000C69'
MQRCCF_NON_FOUND	3200	X'00000C80'
MQRCCF_SECURITY_SWITCH_OFF	3201	X'00000C81'
MQRCCF_SECURITY_REFRESH_FAILED	3202	X'00000C82'
MQRCCF_PARM_CONFLICT	3203	X'00000C83'
MQRCCF_COMMANDE_INHIBÉE	3204	X'00000C84'
MQRCCF_OBJET_BEING_DELETED	3205	X'00000C85'
MQRCCF_STORAGE_CLASS_IN_USE	3207	X'00000C87'
MQRCCF_NOM_OBJET_RESTREINT	3208	X'00000C88'
MQRCCF_OBJET_LIMITE_EXCEEDED	3209	X'00000C89'
MQRCCF_OBJET_OPEN_FORCE	3210	X'00000C8A'

Tableau 305. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRCCF_DISPOSITION_CONFLICT	3211	X'00000C8B'
MQRCCF_Q_MGR_NOT_IN_QSG	3212	X'00000C8C'
MQRCCF_ATTR_VALEUR_FIXE	3213	X'00000C8D'
MQRCCF_NOMELIST_ERREUR	3215	X'00000C8F'
MQRCCF_NO_CHANNEL_INITIATOR	3217	X'00000C91'
MQRCCF_CHANNEL_INITIATOR_ERROR	3218	X'00000C92'
MQRCCF_COMMAND_NIVEAU_CONFLIT	3222	X'00000C96'
MQRCCF_Q_ATTR_CONFLIT	3223	X'00000C97'
MQRCCF_EVENTS_DISABLED	3224	X'00000C98'
MQRCCF_COMMAND_SCOPE_ERROR	3225	X'00000C99'
MQRCCF_COMMAND_REPLY_ERROR	3226	X'00000C9A'
MQRCCF_FONCTION_RESTREINTE	3227	X'00000C9B'
MQRCCF_PARM_MISSING	3228	X'00000C9C'
MQRCCF_PARM_VALEUR_ERREUR	3229	X'00000C9D'
MQRCCF_COMMAND_LENGTH_ERROR	3230	X'00000C9E'
MQRCCF_COMMANDE_ORIGINE_ERREUR	3231	X'00000C9F'
MQRCCF_LISTENER_CONFLICT	3232	X'00000CA0'
MQRCCF_LISTENER_STARTED	3233	X'00000CA1'
MQRCCF_LISTENER_STOPPED	3234	X'00000CA2'
MQRCCF_CHANNEL_ERROR	3235	X'00000CA3'
MQRCCF_CF_STRUC_ERROR	3236	X'00000CA4'
MQRCCF_ID_UTILISATEUR_INCONNU	3237	X'00000CA5'
MQRCCF_ERREUR_INATTENDU	3238	X'00000CA6'
MQRCCF_NO_XCF_PARTNER	3239	X'00000CA7'
MQRCCF_CFGR_PARM_ID_ERROR	3240	X'00000CA8'
MQRCCF_CFIF_LENGTH_ERROR	3241	X'00000CA9'
MQRCCF_CFIF_OPERATOR_ERROR	3242	X'00000CAA'
MQRCCF_CFIF_PARM_ID_ERROR	3243	X'00000CAB'
MQRCCF_CFSF_FILTER_VAL_LEN_ERR	3244	X'00000CAC'
MQRCCF_CFSF_LENGTH_ERROR	3245	X'00000CAD'
MQRCCF_CFSF_ERREUR_OPERATEUR	3246	X'00000CAE'
MQRCCF_CFSF_PARM_ID_ERROR	3247	X'00000CAF'
MQRCCF_TOO_MANY_FILTERS	3248	X'00000CB0'
MQRCCF_LISTENER_RUNNING	3249	X'00000CB1'
MQRCCF_LSTR_STATUS_NOT_FOUND	3250	X'00000CB2'
MQRCCF_SERVICE_RUNNING	3251	X'00000CB3'
MQRCCF_SERV_STATUS_NOT_FOUND	3252	X'00000CB4'
MQRCCF_SERVICE_STOPPED	3253	X'00000CB5'
MQRCCF_CFBS_DUPLICATE_PARM	3254	X'00000CB6'



Tableau 305. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQRCCF_CFBS_LENGTH_ERROR	3255	X'00000CB7'
MQRCCF_CFBS_PARM_ID_ERROR	3256	X'00000CB8'
MQRCCF_CFBS_STRING_LENGTH_ERR	3257	X'00000CB9'
MQRCCF_CFGR_LENGTH_ERROR	3258	X'00000CBA'
MQRCCF_CFGR_PARM_COUNT_ERROR	3259	X'00000CBB'
MQRCCF_CONN_NOT_STOPPED	3260	X'00000CBC'
MQRCCF_SERVICE_REQUEST_PENDING	3261	X'00000CBD'
MQRCCF_NO_START_CMD	3262	X'00000CBE'
MQRCCF_NO_STOP_CMD	3263	X'00000CBF'
MQRCCF_CFBF_LENGTH_ERROR	3264	X'00000CC0'
MQRCCF_CFBF_PARM_ID_ERROR	3265	X'00000CC1'
MQRCCF_CFBF_OPERATOR_ERROR	3266	X'00000CC2'
MQRCCF_CFBF_FILTER_VAL_LEN_ERR	3267	X'00000CC3'
MQRCCF_LISTENER_STILL_ACTIF	3268	X'00000CC4'
MQRCCF_DEF_XMIT_Q_CLUS_ERROR	3269	X'00000CC5'
MQRCCF_TOPICSTR_ALREADY_EXISTS	3300	X'00000CE4'
MQRCCF_SHARING_CONVS_ERROR	3301	X'00000CE5'
MQRCCF_SHARING_CONVS_TYPE	3302	X'00000CE6'
MQRCCF_SECURITY_CASE_CONFLICT	3303	X'00000CE7'
MQRCCF_TOPIC_TYPE_ERROR	3305	X'00000CE9'
MQRCCF_MAX_INSTANCES_ERROR	3306	X'00000CEA'
MQRCCF_MAX_INSTS_PER_CLNT_ERR	3307	X'00000CEB'
MQRCCF_TOPIC_STRING_NOT_FOUND	3308	X'00000CEC'
MQRCCF_SUBSCRIPTION_POINT_ERR	3309	X'00000CED'
MQRCCF_SUB_ALREADY_EXISTS	3311	X'00000CEF'
MQRCCF_NOM_OBJET_INCONNU	3312	X'00000CF0'
MQRCCF_REMOTE_Q_NAME_ERROR	3313	X'00000CF1'
MQRCCF_DURABILITY_NOT_ALLOWED	3314	X'00000CF2'
MQRCCF_HOBJ_ERREUR	3315	X'00000CF3'
MQRCCF_DEST_NAME_ERROR	3316	X'00000CF4'
MQRCCF_DESTINATION_NON_VALIDE	3317	X'00000CF5'
MQRCCF_PUBSUB_INHIBÉ	3318	X'00000CF6'
MQRCCF_CHLAUTH_TYPE_ERREUR	3326	X'00000CFE'
MQRCCF_CHLAUTH_ACTION_ERROR	3327	X'00000CFF'
MQRCCF_CHLAUTH_USERSRC_ERROR	3335	X'00000D07'
MQRCCF_TYPE_AUTH	3336	X'00000D08'
MQRCCF_CHLAUTH_ALREADY_EXISTS	3337	X'00000D09'
MQRCCF_CHLAUTH_NOT_FOUND	3338	X'00000D0A'
MQRCCF_FAUT_CHLAUTH_ACTION	3339	X'00000D0B'


Tableau 305. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRCCF_FAUT_CHLAUTH_USERSRC	3340	X'00000D0C'
MQRCCF_CHLAUTH_WARN_ERREUR	3341	X'00000D0D'
MQRCCF_CONCORDANCE-erreur-chlorauth_correspondance	3342	X'00000D0E'
MQRCCF_IPADDR_RANGE_CONFLIT	3343	X'00000D0F'
MQRCCF_CHLAUTH_MAX_DEPASSE	3344	X'00000D10'
MQRCCF_IPADDR_ERREUR	3345	X'00000D11'
MQRCCF_IPADDR_RANGE_ERROR	3346	X'00000D12'
MQRCCF_NOM_PROFIL_MANQUANT	3347	X'00000D13'
MQRCCF_CHLAUTH_CLNTUSER_ERROR	3348	X'00000D14'
MQRCCF_CHLAUTH_NOM_ERREUR	3349	X'00000D15'
MQRCCF_SUITE_B_ERREUR	3353	X'00000D19'
MQRCCF_PSCLUS_DISABLED_TOPDEF	3359	X'00000D1F'
MQRCCF_PSCLUS_TOPIC_EXISTS	3360	X'00000D20'
MQRCCF_PROTOCOLE_NON_VALIDE	3365	X'00000D25'
MQRCCF_ACCÈS_BLOQUÉ	3382	X'00000D36'
MQRCCF_OBJET_ALREADY_EXISTS	4001	X'00000FA1'
MQRCCF_OBJET_TYPE_ERREUR	4002	X'00000FA2'
MQRCCF_LIKE_OBJECT_MAL_TYPE	4003	X'00000FA3'
MQRCCF_OBS_OPEN	4004	X'00000FA4'
MQRCCF_ATTR_VALEUR_ERREUR	4005	X'00000FA5'
MQRCCF_UNKNOWN_Q_MGR	4006	X'00000FA6'
MQRCCF_Q_TYPE_ERREUR	4007	X'00000FA7'
MQRCCF_OBJECT_NAME_ERROR	4008	X'00000FA8'
MQRCCF_ALLOCATE_FAILED	4009	X'00000FA9'
MQRCCF_HÔTE_NON_DISPONIBLE	4010	X'00000FAA'
MQRCCF_ERREUR de configuration	4011	X'00000FAB'
MQRCCF_CONNECTION_REFUSE	4012	X'00000FAC'
MQRCCF_ENTRY_ERROR	4013	X'00000FAD'
Echec de MQRCCF_SEND_FAILED	4014	X'00000FAE'
MQRCCF_ERREUR_DONNÉES_REÇUES	4015	X'00000FAF'
Echec de MQRCCF_RECEIVE_FAILED	4016	X'00000FB0'
MQRCCF_CONNEXION_FERMÉ	4017	X'00000FB1'
MQRCCF_NO_STORAGE	4018	X'00000FB2'
MQRCCF_NO_COMMS_MANAGER	4019	X'00000FB3'
MQRCCF_LISTENER_NOT_STARTED	4020	X'00000FB4'
Echec de MQRCCF_BIND_FAILED	4024	X'00000FB8'
MQRCCF_CHANNEL_EN attente de validation	4025	X'00000FB9'
MQRCCF_MQCONN_FAILED	4026	X'00000FBA'
MQRCCF_MQOPEN_FAILED	4027	X'00000FBB'

Tableau 305. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRCCF_MQGET_FAILED	4028	X'00000FBC'
Echec de MQRCCF_MQPUT_FAILED	4029	X'00000FBD'
MQRCCF_ERREUR_PING_ERROR	4030	X'00000FBE'
MQRCCF_CHANNEL_IN_USE	4031	X'00000FBF'
MQRCCF_CHANNEL_NOT_FOUND	4032	X'00000FC0'
MQRCCF_UNKNOWN_CANAL_DISTANT	4033	X'00000FC1'
MQRCCF_REMOTE_QM_UNAVAILABLE	4034	X'00000FC2'
MQRCCF_REMOTE_QM_TERMINAISON	4035	X'00000FC3'
MQRCCF_MQINQ_ÉCHEC	4036	X'00000FC4'
MQRCCF_NOT_XMIT_Q	4037	X'00000FC5'
MQRCCF_CHANNEL_DISABLED	4038	X'00000FC6'
MQRCCF_USER_EXIT_NOT_AVAILABLE	4039	X'00000FC7'
Echec de MQRCCF_COMMIT_FAILED	4040	X'00000FC8'
MQRCCF_TYPE_CANAL_INJUSTE	4041	X'00000FC9'
MQRCCF_CHANNEL_ALREADY_EXISTS	4042	X'00000FCA'
MQRCCF_DONNÉES_TOO_LARGE	4043	X'00000FCB'
MQRCCF_CHANNEL_NAME_ERROR	4044	X'00000FCC'
MQRCCF_XMIT_Q_NAME_ERROR	4045	X'00000FCD'
MQRCCF_MCA_NAME_ERROR	4047	X'00000FCF'
MQRCCF_SEND_EXIT_NAME_ERROR	4048	X'00000FD0'
MQRCCF_SEC_EXIT_NAME_ERROR	4049	X'00000FD1'
MQRCCF_MSG_EXIT_NAME_ERROR	4050	X'00000FD2'
MQRCCF_RCV_NOM_EXIT	4051	X'00000FD3'
MQRCCF_XMIT_Q_NAME_MAL_TYPE	4052	X'00000FD4'
MQRCCF_MCA_NOM_TYPE_ERREUR	4053	X'00000FD5'
MQRCCF_DISC_INT_TYPE	4054	X'00000FD6'
MQRCCF_SHORT_RETRY_MAL_TYPE	4055	X'00000FD7'
MQRCCF_SHORT_TIMER_MAL_TYPE	4056	X'00000FD8'
MQRCCF_LONG_RETRY_TYPE	4057	X'00000FD9'
MQRCCF_LONG_TIMER_TYPE	4058	X'00000FDA'
MQRCCF_PUT_AUTH_TYPE_ERREUR	4059	X'00000FDB'
MQRCCF_KEEP_ALIVE_INT_ERROR	4060	X'00000FDC'
MQRCCF_NOM_CONN_MANQUANT	4061	X'00000FDD'
MQRCCF_CONN_NAME_ERROR	4062	X'00000FDE'
MQRCCF_MQSET_FAILED	4063	X'00000FDF'
MQRCCF_CHANNEL_NOT_ACTIVE	4064	X'00000FE0'
MQRCCF_TERMINÉ_PAR_EXIT	4065	X'00000FE1'
MQRCCF_DYNAMIC_Q_SCOPE_ERROR	4067	X'00000FE3'
MQRCCF_CELL_DIR_NOT_AVAILABLE	4068	X'00000FE4'

Tableau 305. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRCCF_MR_COUNT_ERROR	4069	X'00000FE5'
MQRCCF_MR_COUNT_WRONG_TYPE	4070	X'00000FE6'
MQRCCF_MR_EXIT_NAME_ERROR	4071	X'00000FE7'
MQRCCF_MR_EXIT_NAME_WRONG_TYPE	4072	X'00000FE8'
MQRCCF_MR_INTERVAL_ERREUR	4073	X'00000FE9'
MQRCCF_MR_INTERVAL_TYPE_ERREUR	4074	X'00000FEA'
MQRCCF_NPM_SPEED_ERROR	4075	X'00000FEB'
MQRCCF_NPM_SPEED_INJUSTEMENT_type	4076	X'00000FEC'
ERREUR MQRCCF_HB_INTERVAL_ERROR	4077	X'00000FED'
MQRCCF_HB_INTERVAL_TYPE_ERREUR	4078	X'00000FEE'
MQRCCF_CHAD_ERREUR	4079	X'00000FEF'
MQRCCF_CHAD_TYPE_ERREUR	4080	X'00000FF0'
ERREUR MQRCCF_CHAD_EVENT_ERROR	4081	X'00000FF1'
MQRCCF_CHAD_EVENT_MAL_TYPE	4082	X'00000FF2'
MQRCCF_CHAD_EXIT_ERREUR	4083	X'00000FF3'
MQRCCF_CHAD_EXIT_TYPE_ERREUR	4084	X'00000FF4'
MQRCCF_SUPPRESSED_BY_EXIT	4085	X'00000FF5'
MQRCCF_BATCH_INT_ERREUR	4086	X'00000FF6'
MQRCCF_BATCH_INT_TYPE	4087	X'00000FF7'
MQRCCF_NET_PRIORITY_ERROR	4088	X'00000FF8'
MQRCCF_NET_PRIORITÉ_TYPE_ERREUR	4089	X'00000FF9'
MQRCCF_CHANNEL_CLOSED	4090	X'00000FFA'
MQRCCF_Q_STATUS_NOT_FOUND	4091	X'00000FFB'
MQRCCF_SSL_CIPHER_SPEC_ERROR	4092	X'00000FFC'
MQRCCF_SSL_PEER_NAME_ERROR	4093	X'00000FFD'
MQRCCF_SSL_CLIENT_AUTH_ERROR	4094	X'00000FFE'
MQRCCF_RETAINED_NOT_SUPPORTED	4095	X'00000FFF'
 MQRCCF_KWD_VALUE_TYPE_ERREUR	4096	X'00001000'

### MQRCN\_\* (Constantes de reconnexion client)

Tableau 306. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRCN_NO	0	X'00000000'
MQRCN_YES	1	X'00000001'
MQRCN_Q_MGR	2	X'00000002'
MQRCN_DISABLED	3	X'00000003'

## MQRCVTIME\_\* (types de délai d'attente de réception)

Tableau 307. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRCVTIME_MULTIPLIER	0	X'00000000'
MQRCVTIME_ADD	1	X'00000001'
MQRCVTIME_EQUAL	2	X'00000002'

## MQREADA\_\* (valeurs de lecture anticipée)

Tableau 308. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQREADA_NO	0	X'00000000'
MQREADA_YES	1	X'00000001'
MQREADA_DISABLED	2	X'00000002'
MQREADA_INHIBÉ	3	X'00000003'
MQREADA_BACKLOG	4	X'00000004'

## MQRECORDING\_\* (options d'enregistrement)

Tableau 309. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRECORDING_DISABLED	0	X'00000000'
MQRECORDING_Q	1	X'00000001'
MSG_ENREGISTREMENT_MQ	2	X'00000002'

## MQREGO\_\* (Options d'enregistrement de publication / abonnement)

Tableau 310. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQREGO_AUCUN	0	X'00000000'
MQREGO_CORREL_ID_AS_IDENTITY	1	X'00000001'
MQREGO_ANONYME	2	X'00000002'
MQREGO_LOCAL	4	X'00000004'
MQREGO_DIRECT_REQUESTS	8	X'00000008'
MQREGO_NEW_PUBLICATIONS_ONLY	16	X'00000010'
MQREGO_PUBLISH_ON_REQUEST_ONLY	32	X'00000020'
MQREGO_DEREGISTER_ALL	64	X'00000040'
MQREGO_INCLUDE_STREAM_NAME	128	X'00000080'
MQREGO_INFORM_SI_RETENU	256	X'00000100'
MQREGO_DUPLICATES_OK	512	X'00000200'
MQREGO_NON_PERSISTENT	1024	X'00000400'
MQREGO_PERSISTENT	2048	X'00000800'
MQREGO_PERSISTENT_AS_PUBLISH	4096	X'00001000'
MQREGO_XX_ENCODE_CASE_ONE persistent_AS_q	8192	X'00002000'

Tableau 310. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
NOM_ADD_MQREGO_	16384	X'00004000'
MQREGO_NO_ALTÉRATION	32768	X'00008000'
MQREGO RÉPONSE_COMPLET RÉPONSE	65536	X'00010000'
MQREGO_JOIN_SHARED	131072	X'00020000'
MQREGO_JOIN_EXCLUSIVE	262144	X'00040000'
MQREGO_LEAVE_ONLY	524288	X'00080000'
ID_UTILISATEUR_MQREGO_VARIABLE_	1048576	X'00100000'
MQREGO_VERROUILLÉ	2097152	X'00200000'

## MQRFH\_\* (règles et mise en forme de la structure d'en-tête et des indicateurs)

### Règles et structure d'en-tête de formatage

Tableau 311. Structures de constantes	
Nom	Structure
ID MQRFH_STRUC_ID	"RFH¬"
MQRFH_STRUC_ID_ARRAY	'R', 'F', 'H', '¬'

**Remarque :** Le symbole ¬ représente un caractère blanc unique.

Tableau 312. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQRFH_VERSION_1	1	X'00000001'
MQRFH_VERSION_2	2	X'00000002'
MQRFH_STRUC_LENGTH_FIXED	32	X'00000020'
MQRFH_STRUC_LENGTH_FIXED_2	36	X'00000024'

### Règles et indicateurs d'en-tête de formatage

Tableau 313. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQRFH_AUCUN	0	X'00000000'
MQRFH_NO_FLAGS	0	X'00000000'

## MQRFH2\_\* (balise Options de publication / abonnement RFH2 Balises de dossier de niveau supérieur)

Tableau 314. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQRFH2_NAME_VALUE_VERSION	1	X'00000001'

### MQRFH2\_\* (noms de balise d'options de publication / abonnement)

MQRFH2_PUBSUB_CMD_FOLDER	"psc"
--------------------------	-------

MQRFH2_PUBSUB_RESP_FOLDER	"pscɾ"
MQRFH2_MSG_CONTENT_FOLDER	"mcd"
MQRFH2_USER_FOLDER	"usr"

### ***MQRFH2\_\* (noms de balise XML des options de publication / abonnement)***

MQRFH2_PUBSUB_CMD_FOLDER_B	"<psc>"
MQRFH2_PUBSUB_CMD_FOLDER_E	"</psc>"
MQRFH2_PUBSUB_RESP_FOLDER_B	"<pscɾ>"
MQRFH2_PUBSUB_RESP_FOLDER_E	"</pscɾ>"
MQRFH2_MSG_CONTENT_FOLDER_B	"<mcd>"
MQRFH2_MSG_CONTENT_FOLDER_E	"</mcd>"
MQRFH2_USER_FOLDER_B	"<usr>"
MQRFH2_USER_FOLDER_E	"</usr>"

### **MQRL\_\* (Longueur renvoyée)**

<i>Tableau 315. Valeurs des constantes</i>		
Nom	Valeur décimale	Valeur hexadécimale
MQRL_UNDEFINED	-1	X'FFFFFFFF'

### **MQRMH\_\* (Structure d'en-tête de message de référence)**

<i>Tableau 316. Structures de constantes</i>	
Nom	Structure
ID_STRUC_MQRMH_	"RMH↵"
MQRMH_STRUC_ID_ARRAY	'R', 'M', 'H', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

<i>Tableau 317. Valeurs des constantes</i>		
Nom	Valeur décimale	Valeur hexadécimale
MQRMH_VERSION_1	1	X'00000001'
MQRMH_VERSION_CURRENT_VERSION	1	X'00000001'

### **MQRMHF\_\* (indicateurs d'en-tête de message de référence)**

<i>Tableau 318. Valeurs des constantes</i>		
Nom	Valeur décimale	Valeur hexadécimale
MQRMHF_LAST	1	X'00000001'
MQRMHF_NOT_LAST	0	X'00000000'

## MQRO\_\* (options de rapport)

Tableau 319. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRO_EXCEPTION	16777216	X'01000000'
MQRO_EXCEPTION_WITH_DATA	50331648	X'03000000'
MQRO_EXCEPTION_WITH_FULL_DATA	117440512	X'07000000'
MQRO_EXPIRATION	2097152	X'00200000'
MQRO_EXPIRATION_WITH_DATA	6291456	X'00600000'
MQRO_EXPIRATION_WITH_FULL_DATA	14680064	X'00E00000'
MQRO_COA	256	X'00000100'
MQRO_COA_WITH_DATA	768	X'00000300'
MQRO_COA_WITH_FULL_DATA	1792	X'00000700'
MQRO_COD	2048	X'00000800'
MQRO_COD_WITH_DATA	6144	X'00001800'
MQRO_COD_WITH_FULL_DATA	14336	X'00003800'
MQRO_PAN	1	X'00000001'
MQRO_NAN	2	X'00000002'
ACTIVITE MQRO	4	X'00000004'
MQRO_NEW_MSG_ID	0	X'00000000'
MQRO_PASS_MSG_ID	128	X'00000080'
MQRO_COPY_MSG_ID_TO_CORREL_ID	0	X'00000000'
MQRO_PASS_CORREL_ID	64	X'00000040'
MQRO_DEAD_LETTER_Q	0	X'00000000'
MQRO_DISCARD_MSG	134217728	X'08000000'
MQRO_PASS_DISCARD_ET_EXPIRATION	16384	X'00004000'
MQRO_AUCUN	0	X'00000000'

## MQRO\_\* (Masques d'options de rapport)

Tableau 320. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRO_REJECT_UNSUP_MASK	270270464	X'101C0000'
MQRO_ACCEPT_UNSUP_MASK	-270532353	X'EFE000FF'
MQRO_ACCEPT_UNSUP_IF_XMIT_MASK	261888	X'0003FF00'

## MQROUTE\_\* (Trace-route)

### Nombre maximal d'activités de routage de trace (MQIACF\_MAX\_ACTIVITÉS)

Tableau 321. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQROUTE_UNLIMITED_ACTIVITÉS	0	X'00000000'



## Détails du routage de trace (MQIACF\_ROUTE\_DETAIL)

Tableau 322. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQROUTE_DETAIL_LOW	2	X'00000002'
MQROUTE_DETAIL_MEDIUM	8	X'00000008'
MQROUTE_DETAIL_ELEVE	32	X'00000020'

## Acheminement de la route de trace (MQIACF\_ROUTE\_FORWARDING)

Tableau 323. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQROUTE_ACHEMINEMENT_TOUT	256	X'00000100'
MQROUTE_WARD_IF_SUPPORTED	512	X'00000200'
MQROUTE_TRANSFER_REJ_UNSUP_MASK	-65536	X'FFFF0000'

## Distribution de routage de trace (MQIACF\_ROUTE\_DELIVERY)

Tableau 324. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQROUTE_DELIVER_YES	4096	X'00001000'
MQROUTE_NO	8192	X'00002000'
MQROUTE_LIVRE_REJ_UNSUP_MASK	-65536	X'FFFF0000'

## Accumulation de route de trace (MQIACF\_ROUTE\_ACCUMULATION)

Tableau 325. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQROUTE_ACCUMULATE_NONE	65539	X'00010003'
MQROUTE_ACCUMULATE_EN_MSG	65540	X'00010004'
MQROUTE_ACCUMULATE_ET_REPONSE	65541	X'00010005'

## MQRP\_\* (options de remplacement de format de commande)

Tableau 326. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQRP_OUI	1	X'00000001'
MQRP_NON	0	X'00000000'

## MQRQ\_\* (Qualificateurs de raison de format de commande)

Tableau 327. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQRQ_CONN_NOT_AUTHORIZED	1	X'00000001'
MQRQ_OPEN_NOT_AUTHORIZED	2	X'00000002'
MQRQ_CLOSE_NOT_AUTHORIZED	3	X'00000003'
MQRQ_CMD_NOT_AUTHORIZED	4	X'00000004'

Tableau 327. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQRQ_Q_MGR_STOPPING	5	X'00000005'
MQRQ_Q_MGR QUIESCING	6	X'00000006'
MQRQ_CHANNEL_STOPPED_OK	7	X'00000007'
MQRQ_CHANNEL_STOPPED_ERROR	8	X'00000008'
MQRQ_CHANNEL_STOPPED_RETRY	9	X'00000009'
MQRQ_CHANNEL_STOPPED_DISABLED	10	X'0000000A'
MQRQ_BRIDGE_STOPPED_OK	11	X'0000000B'
MQRQ_BRIDGE_STOPPED_ERROR	12	X'0000000C'
MQRQ_SSL_HANDSHAKE_ERREUR	13	X'0000000D'
MQRQ_SSL_CIPHER_SPEC_ERROR	14	X'0000000E'
MQRQ_SSL_CLIENT_AUTH_ERROR	15	X'0000000F'
MQRQ_SSL_PEER_NAME_ERREUR	16	X'00000010'
MQRQ_SUB_NOT_AUTHORIZED	17	X'00000011'
MQRQ_SUB_DEST_NOT_AUTHORIZED	18	X'00000012'
MQRQ_SSL_UNKNOWN_RÉVOCATION	19	X'00000013'
MQRQ_SYS_CONN_NOT_AUTHORIZED	20	X'00000014'
ADRESSE_BLOC_CANAL_MQRQ_CANAL	21	X'00000015'
MQRQ_CHANNEL_BLOCKED_USERID	22	X'00000016'
MQRQ_CHANNEL_BLOCKED_NOACCESS	23	X'00000017'
MQRQ_MAX_ACTIVE_CHANNELS	24	X'00000018'
MQRQ_MAX_CHANNELS	25	X'00000019'
MQRQ_SVRCONN_INST_LIMIT	26	X'0000001A'
MQRQ_CLIENT_INST_LIMIT !	27	X'0000001B'
MQRQ_CAF_NOT_INSTALLED	28	X'0000001C'
MQRQ_CSP_NOT_AUTHORIZED	29	X'0000001D'
MQRQ_FAILOVER_PERMITTED	30	X'0000001E'
MQRQ_FAILOVER_NOT_PERMITTED	31	X'0000001F'
MQRQ_STANDBY_ACTIVEE	32	X'00000020'
MQRQ_REPLICA_ACTIVÉ	33	X'00000021'

### MQRT\_\* (Types d'actualisation de format de commande)

Tableau 328. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
CONFIGURATION_MQR	1	X'00000001'
EXPIRATION_MQRT_EXPIRATION	2	X'00000002'
MQRT_NSPROC	3	X'00000003'
MQRT_PROXYSUB	4	X'00000004'

## MQRU\_\* (Demande uniquement)

Tableau 329. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQRU_PUBLISH_ON_REQUEST	1	X'00000001'
MQRU_PUBLISH_ALL	2	X'00000002'

## MQSCA\_\* (authentification de client TLS)

Tableau 330. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSCA_REQUIS	0	X'00000000'
MQSCA_XX_ENCODE_CASE_ONE facultatif	1	X'00000001'

## MQSCO\_\* (options de configuration TLS)

### Structure des options de configuration TLS

Tableau 331. Structures de constantes	
Nom	Structure
ID_STRUC_MQSCO_	"SCO↵"
MQSCO_STRUC_ID_ARRAY	'S', 'C', 'O', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 332. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSCO_VERSION_1	1	X'00000001'
MQSCO_VERSION_2	2	X'00000002'
MQSCO_VERSION_3	3	X'00000003'
MQSCO_VERSION_4	4	X'00000004'
VERSION MQSCO_CURRENT_VERSION	4	X'00000004'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

### Nombre de réinitialisations de clé des options de configuration TLS

Tableau 333. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSCO_RESET_COUNT_DEFAULT	0	X'00000000'

### Portée des définitions de file d'attente de format de commande

Tableau 334. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSCO_Q_DIR	1	X'00000001'
MQSCO_CELL	2	X'00000002'

## MQSCOPE\_\* (portée de publication)

Tableau 335. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSCOPE_ALL	0	X'00000000'
MQSCOPE_AS_PARENT	1	X'00000001'
MQSCOPE_QMGR	4	X'00000004'

## MQSCYC\_\* (cas de sécurité)

Tableau 336. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSCYC_SUPÉRIEURE	0	X'00000000'
MQSCYC_MIXTE	1	X'00000001'

## MQSD\_\* (structure de descripteur d'objet)

Tableau 337. Noms et structures des constantes

Nom	Structure
ID_STRUCD_MQS	"SD↵↵"
MQSD_STRUC_ID_ARRAY	'S','D','↵','↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 338. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSD_VERSION_1	1	X'00000001'
MQSD_CURRENT_VERSION	1	X'00000001'

## MQSECITEM\_\* (Éléments de sécurité de format de commande)

Tableau 339. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSECITEM_ALL	0	X'00000000'
MQSECITEM_MQADMIN	1	X'00000001'
MQSECITEM_MQNLIST	2	X'00000002'
MQSECITEM_MQPROC	3	X'00000003'
MQSECITEM_MQQUEUE	4	X'00000004'
MQSECITEM_MQCONN	5	X'00000005'
MQSECITEM_MQCMLS	6	X'00000006'
MQSECITEM_MXADMIN	7	X'00000007'
MQSECITEM_MXNLIST	8	X'00000008'
MQSECITEM_MXPROC	9	X'00000009'
MQSECITEM_MXQUEUE	10	X'0000000A'
MQSECITEM_MXTOPIC	11	X'0000000B'

## MQSECPROT\_\* (Types de protocole de sécurité)

Tableau 340. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSECPROT_AUCUN	0	X'00000000'
MQSECPROT_SSLV30	1	X'00000001'
MQSECPROT_TLSV10	2	X'00000002'
MQSECPROT_TLSV12	4	X'00000004'

## MQSECSW\_\* (Commutateurs de sécurité de format de commande et états de commutation)

### Commutateurs de sécurité de format de commande

Tableau 341. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSECSW_PROCESSUS	1	X'00000001'
MQSECSW_NAMELIST	2	X'00000002'
MQSECSW_Q	3	X'00000003'
MQSECSW_TOPIC	4	X'00000004'
MQSECSW_CONTEXT	6	X'00000006'
MQSECSW_ALTERNATE_USER	7	X'00000007'
MQSECSW_COMMAND	8	X'00000008'
MQSECSW_CONNECTION	9	X'00000009'
MQSECSW_SOUS-SYSTEME	10	X'0000000A'
RESSOURCES MQSECSW_COMMAND_RESSOURCES	11	X'0000000B'
MQSECSW_Q_DIR	15	X'0000000F'
MQSECSW_QSG	16	X'00000010'

### Etats des commutateurs de sécurité de format de commande

Tableau 342. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSECSW_OFF_FOUND	21	X'00000015'
MQSECSW_ON_FOUND	22	X'00000016'
MQSECSW_OFF_NOT_FOUND	23	X'00000017'
MQSECSW_ON_NOT_FOUND	24	X'00000018'
ERREUR MQSECSW_OFF_ERROR	25	X'00000019'
MQSECSW_ON_SUBSTITUÉ	26	X'0000001A'

## MQSECTYPE\_\* (Types de sécurité de format de commande)

Tableau 343. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSECTYPE_AUTHSERV	1	X'00000001'

Tableau 343. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQSECTYPE_SSL	2	X'00000002'
MQSECTYPE_CLASSES	3	X'00000003'

## MQSEG\_\* (Segmentation)

Tableau 344. Noms et valeurs de constante	
Nom	Valeur
MQSEG_INHIBÉ	'↵'
MQSEG_AUTORISÉ_	'A'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

## MQSEL\_\* (Valeurs de sélecteur spéciales)

Tableau 345. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSEL_ANY_SELECTOR	-30001	X'FFFF8ACF'
MQSEL_ANY_USER_SELECTOR	-30002	X'FFFF8ACE'
MQSEL_ANY_SYSTEM_SELECTOR	-30003	X'FFFF8ACD'
MQSEL_ALL_SELECTEURS	-30001	X'FFFF8ACF'
MQSEL_ALL_USER_SELECTEURS	-30002	X'FFFF8ACE'
MQSEL_ALL_SYSTEM_SELECTEURS	-30003	X'FFFF8ACD'

## MQSELTYPE\_\* (types de sélecteur)

Tableau 346. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSELTYPE_NONE	0	X'00000000'
MQSELTYPE_STANDARD	1	X'00000001'
MQSELTYPE_ETENDU	2	X'00000002'

## MQSID\_\* (identificateur de sécurité)

Tableau 347. Noms et valeurs de constante	
Nom	Valeur
MQSID_AUCUN	X'00...00' (40 valeurs nulles)
MQSID_NON_ARRAY	'\0', '\0', ... (40 valeurs nulles)

## MQSIDT\_\* (types d'identificateur de sécurité)

Tableau 348. Noms et valeurs de constante	
Nom	Valeur hexadécimale
MQSIDT_AUCUN	X'00'
ID_SÉCURITÉ_NT_MQSIDT_ID	X'01'
MQSIDT_ID_SÉCURITÉ_WAS	X'02'

## MQSMPO\_\* (Définition des options et de la structure des propriétés de message)

### Définir la structure des options de propriété de message

Tableau 349. Structures de constantes

Nom	Structure
ID_STRUC_MQSMPO_	"SMPO"
MQSMPO_STRUC_ID_ARRAY	'S', 'M', 'P', 'O'

**Remarque :** Le symbole – représente un caractère blanc unique.

Tableau 350. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSMPO_VERSION_1	1	X'00000001'
MQSMPO_CURRENT_VERSION	1	X'00000001'

### Définir les options de propriété de message

Tableau 351. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSMPO_SET_FIRST	0	X'00000000'
MQSMPO_SET_PROP_UNDER_CURSOR	1	X'00000001'
MQSMPO_SET_PROP_AFTER_CURSOR	2	X'00000002'
MQSMPO_XX_ENCODE_CASE_ONE annexes	4	X'00000004'
MQSMPO_SET_PROP_BEFORE_CURSOR	8	X'00000008'
MQSMPO_NONE	0	X'00000000'

## MQSO\_\* (options d'abonnement)

Tableau 352. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSO_AUCUN	0	X'00000000'
MQSO_NON_DURABLE	0	X'00000000'
MQSO_READ_AHEAD_AS_Q_DEF	0	X'00000000'
MQSO_ALTER	1	X'00000001'
MQSO_CREER	2	X'00000002'
MQSO_RESUME	4	X'00000004'
MQSO_DURABLE	8	X'00000008'
SOUS-GROUPE_MQSO	16	X'00000010'
MQSO_GERE	32	X'00000020'
MQSO_SET_IDENTITY_CONTEXT	64	X'00000040'
ID_UTILISATEUR_FIXE_MQSO_FIXE	256	X'00000100'
ID utilisateur MQSO_ANY_USERID	512	X'00000200'
MQSO_PUBLICATIONS_ON_REQUEST	2048	X'00000800'
MQSO_NEW_PUBLICATIONS_ONLY	4096	X'00001000'

Tableau 352. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQSO_FAIL_IF QUIESCING	8192	X'00002000'
MQSO_ALTERNATE_USER_AUTHORITY	262144	X'00040000'
MQSO_WILDCARD_CHAR	1048576	X'00100000'
MQSO_WILDCARD_TOPIC	2097152	X'00200000'
ID_CORREL_SET_MQSO	4194304	X'00400000'
MQSO_SCOPE_QMGR	67108864	X'04000000'
MQSO_NO_READ_AHEAD	134217728	X'08000000'
MQSO_READ_AHEAD	268435456	X'10000000'

### MQSP\_\* (disponibilité du point de synchronisation)

Tableau 353. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSP_XX_ENCODE_CASE_ONE disponible	1	X'00000001'
MQSP_NON DISPONIBLE	0	X'00000000'

### MQSPL\_\* (options de protection des règles de sécurité)

Tableau 354. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSPL_PASSTHRU	0	X'00000000'
MQSPL_REMOVE	1	X'00000001'
REGDE_AS_MQSPL	2	X'00000002'

### MQSQQM\_\* (nom du gestionnaire de files d'attente partagées)

Tableau 355. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSQQM_USE	0	X'00000000'
MQSQQM_IGNORE	1	X'00000001'

### MQSR\_\* (Action)

Tableau 356. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
PUBLICATION MQSR_ACTION_PUBLICATION	1	X'00000001'

### MQSRO\_\* (structure des options de demande d'abonnement)

Tableau 357. Structures de constantes	
Nom	Structure
ID_STRUC_MQS	"SR0↵"
MQSRO_STRUC_ID_ARRAY	'S', 'R', 'O', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.



Tableau 358. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSRO_VERSION_1	1	X'00000001'
VERSION_MQSRO_CURRENT_VERSION	1	X'00000001'
MQSRO_AUCUN	0	X'00000000'
MQSRO_FAIL_IF QUIESCING	8192	X'00002000'

## MQSS\_\* (statut du segment)

Tableau 359. Noms et structures des constantes	
Nom	Structure
MQSS_NON_SEGMENT	'-'
SEGMENT_MQSS_SEGMENT	'S'
SEGMENT_LAST_MQSS_	'L'

**Remarque :** Le symbole - représente un caractère blanc unique.

## MQSSL\_\* (exigences TLS FIPS)

**Remarque :** Sous AIX, Linux, and Windows, IBM MQ fournit la conformité à la norme FIPS 140-2 via le module cryptographique IBM Crypto for C (ICC) . Le certificat de ce module a été déplacé vers le statut Historique. Les clients doivent afficher le certificat IBM Crypto for C (ICC) et prendre connaissance des conseils fournis par le NIST. Un module FIPS 140-3 de remplacement est actuellement en cours et son statut peut être affiché en le recherchant dans la [liste des modules NIST CMVP en cours de traitement](#).

Tableau 360. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSSL_FIPS_NO	0	X'00000000'
MQSSL_FIPS_YES	1	X'00000001'

## MQSTAT\_\* (options de statistiques)

Tableau 361. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSTAT_TYPE_ASYNC_ERROR	0	X'00000000'
MQSTAT_TYPE_RECONNEXION	0	X'00000000'
MQSTAT_TYPE_RECONNECTION_ERROR	0	X'00000000'

## MQSTS\_\* (Structure de la structure de génération de rapports de statut)

Tableau 362. Structures de constantes	
Nom	Structure
ID_STRUCTURE_MQST	"STAT"
MQSTS_STRUC_ID_ARRAY	'S', 'T', 'A', 'T'

**Remarque :** Le symbole - représente un caractère blanc unique.

Tableau 363. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSTS_VERSION_1	1	X'00000001'
MQSTS_CURRENT_VERSION	1	X'00000001'

## MQSUB\_\* (Abonnements durables)

### Abonnements durables autorisés

Tableau 364. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSUB_DURABLE_AS_PARENT	0	X'00000000'
MQSUB_DURABLE_ALLOWED	1	X'00000001'
MQSUB_DURABLE_INHIBÉE	2	X'00000002'

### Plage d'abonnements durables

Tableau 365. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSUB_DURABLE_ALL	-1	X'FFFFFFFF'
MQSUB_DURABLE_YES	1	X'00000001'
MQSUB_DURABLE_NO	2	X'00000002'

## MQSUBTYPE\_\* (Types d'abonnement au format de commande)

Tableau 366. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSUBTYPE_API	1	X'00000001'
MQSUBTYPE_ADMIN	2	X'00000002'
MQSUBTYPE_PROXY	3	X'00000003'
MQSUBTYPE_ALL	-1	X'FFFFFFFF'
MQSUBTYPE_UTILISATEUR	-2	X'FFFFFFFE'

## MQSUS\_\* (statut d'interruption du format de commande)

Tableau 367. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSUS_OUI	1	X'00000001'
MQSUS_NO	0	X'00000000'

## MQSVC\_\* (Service)

### Types de service

Tableau 368. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSVC_TYPE_COMMANDE	0	X'00000000'

Tableau 368. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
SERVEUR_TYPE_MQ	1	X'00000001'

### Contrôles de service

Tableau 369. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSVC_CONTROL_Q_MGR	0	X'00000000'
MQSVC_CONTROL_Q_MGR_START	1	X'00000001'
MQSVC_CONTROL_MANUAL	2	X'00000002'

### Statut de service

Tableau 370. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSVC_STATUS_STOPPED	0	X'00000000'
MQSVC_STATUS_DEMARRAGE	1	X'00000001'
MQSVC_STATUS_RUNNING	2	X'00000002'
MQSVC_STATUS_ARRÊT	3	X'00000003'
MQSVC_STATUS_NOUVELLE tentative	4	X'00000004'

### MQSYNCPPOINT\_\* (valeurs de point de synchronisation de format de commande pour la migration de publication / d'abonnement)

Tableau 371. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSYNCPPOINT_OUI	0	X'00000000'
MQSYNCPPOINT_IFPER	1	X'00000001'

### MQSYSP\_\* (valeurs de paramètre système de format de commande)

Tableau 372. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQSYSP_NO	0	X'00000000'
MQSYSP_OUI	1	X'00000001'
MQSYSP_ETENDU	2	X'00000002'
MQSYSP_TYPE_INITIAL	10	X'0000000A'
MQSYSP_TYPE_SET	11	X'0000000B'
MQSYSP_TYPE_LOG_COPY	12	X'0000000C'
ETAT_LOG_TYPE_MQSYSP_	13	X'0000000D'
MQSYSP_TYPE_ARCHIVE_TAPE	14	X'0000000E'
MQSYSP_ALLOC_BLK	20	X'00000014'
MQSYSP_ALLOC_TRK	21	X'00000015'
MQSYSP_ALLOC_CYL	22	X'00000016'

Tableau 372. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQSYSP_STATUS_BUSY	30	X'0000001E'
MQSYSP_STATUS_PREMOUNT	31	X'0000001F'
MQSYSP_STATUS_XX_ENCODE_CASE_ONE disponible	32	X'00000020'
MQSYSP_STATUS_INCONNU	33	X'00000021'
MQSYSP_STATUS_ALLOC_ARCHIVE	34	X'00000022'
MQSYSP_STATUS_COPYING_BSDS	35	X'00000023'
MQSYSP_STATUS_COPYING_LOG	36	X'00000024'

## MQTA\_\* (attributs de rubrique)

### Caractères génériques

Tableau 373. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
BLOC_MQTA_BLOC	1	X'00000001'
MQTA_PASSTHRU	2	X'00000002'

### Abonnements autorisés

Tableau 374. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQTA_SUB_AS_PARENT	0	X'00000000'
MQTA_SUB_INHIBÉ	1	X'00000001'
MQTA_SOUS-AUTORISÉ_AUTORISÉ	2	X'00000002'

### Sous-propagation du proxy

Tableau 375. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQTA_PROXY_SUB_FORCE	1	X'00000001'
MQTA_PROXY_SUB_FIRSTUSE	2	X'00000002'

### Publications autorisées

Tableau 376. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQTA_PUB_AS_PARENT	0	X'00000000'
MQTA_PUB_INHIBÉ	1	X'00000001'
MQTA_PUB_AUTORISÉ	2	X'00000002'

## MQTC\_\* (contrôles de déclencheur)

Tableau 377. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQTC_OFF	0	X'00000000'

Tableau 377. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQTC_ON	1	X'00000001'

### MQTCPKEEP\_\* (signal de présence TCP)

Tableau 378. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQTCPKEEP_NO	0	X'00000000'
MQTCPKEEP_OUI	1	X'00000001'

### MQTCPSTACK\_\* (types de pile TCP)

Tableau 379. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQTCPSTACK_SINGLE	0	X'00000000'
MQTCPSTACK_MULTIPLE	1	X'00000001'

### MQTIME\_\* (Unités de temps de format de commande)

Tableau 380. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQTIME_UNIT_MINS	0	X'00000000'
MQTIME_UNIT_SECS	1	X'00000001'

### MQTM\_\* (structure de message de déclenchement)

Tableau 381. Structures de constantes	
Nom	Structure
ID_STRUC_MQTM	"TM↵"
MQTM_STRUC_ID_ARRAY	'T', 'M', '↵', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 382. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQTM_VERSION_1	1	X'00000001'
MQTM_CURRENT_VERSION	1	X'00000001'

### MQTMC\_\* (structure de format de caractère de message de déclencheur)

Tableau 383. Structures de constantes	
Nom	Structure
ID MQTMC_STRUC_ID	"TMC↵"
MQTMC_STRUC_ID_ARRAY	'T', 'M', 'C', '↵'
MQTMC_VERSION_1	"↵↵1"
MQTMC_VERSION_2	"↵↵2"
MQTMC_CURRENT_VERSION	"↵↵2"

Tableau 383. Structures de constantes (suite)	
Nom	Structure
MQTM_VERSION_1_ARRAY	'-', '-', '-', '1'
MQTM_VERSION_2_ARRAY	'-', '-', '-', '2'
MQTM_CURRENT_VERSION_ARRAY	'-', '-', '-', '2'

### **MQTOPT\_\* (Type de rubrique)**

Tableau 384. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQTOPT_LOCAL	0	X'00000000'
MQTOPT_CLUSTER	1	X'00000001'
MQTOPT_ALL	2	X'00000002'

### **MQTRAXSTR\_\* (Démarrage automatique de la trace de l'initialisateur de canal)**

Tableau 385. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQTRAXSTR_NO	0	X'00000000'
MQTRAXSTR_YES	1	X'00000001'

### **MQTSCOPE\_\* (portée de l'abonnement)**

Tableau 386. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQTSCOPE_QMGR	1	X'00000001'
MQTSCOPE_ALL	2	X'00000002'

### **MQTT\_\* (types de déclencheur)**

Tableau 387. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQTT_AUCUN	0	X'00000000'
MQTT_PREMIER	1	X'00000001'
MQTT_EVERY	2	X'00000002'
MQTT_DEPTH	3	X'00000003'

### **MQTYPE\_\* (types de données de propriété)**

Tableau 388. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQTYPE_AS_SET	0	X'00000000'
MQTYPE_NULL	2	X'00000002'
MQTYPE_BOOLÉEN	4	X'00000004'
MQTYPE_BYTE_STRING	8	X'00000008'
MQTYPE_INT8	16	X'00000010'

Tableau 388. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQTYPE_INT16	32	X'00000020'
MQTYPE_INT32	64	X'00000040'
MQTYPE_LONG	64	X'00000040'
MQTYPE_INT64	128	X'00000080'
MQTYPE_FLOAT32	256	X'00000100'
MQTYPE_FLOAT64	512	X'00000200'
MQTYPE_CHAINE	1024	X'00000400'

### **MQUA\_\* (Sélecteurs d'attribut utilisateur de publication / abonnement)**

Tableau 389. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQUA_FIRST	65536	X'00010000'
MQUA_LAST	99999999	X'3B9AC9FF'

### **MQUIDSUPP\_\* (Prise en charge de l'ID utilisateur au format de commande)**

Tableau 390. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQUIDSUPP_NO	0	X'00000000'
MQUIDSUPP_OUI	1	X'00000001'

### **MQUNDELIVERED\_\* (valeurs de format de commande non distribuées pour la migration de publication / abonnement)**

Tableau 391. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
Mqundelivered_normal	0	X'00000000'
MQUNDELIVERED_SAFE	1	X'00000001'
Carte MQUNDELIVERED_DISCARD	2	X'00000002'
MQUNDELIVERED_KEEP	3	X'00000003'

### **MQUOWST\_\* (Etats de l'unité de travail du format de commande)**

Tableau 392. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQUOWST_AUCUN	0	X'00000000'
MQUOWST_ACTIF	1	X'00000001'
MQUOWST_PREPARE	2	X'00000002'
MQUOWST_NON résolu	3	X'00000003'

## MQUOWT\_\* (Types d'unité de travail de format de commande)

Tableau 393. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQUOWT_Q_MGR	0	X'00000000'
MQUOWT_CICS	1	X'00000001'
RRS MQUOWT_RRS	2	X'00000002'
MQUOWT_IMS	3	X'00000003'
MQUOWT_XA	4	X'00000004'

## MQUS\_\* (utilisations de file d'attente)

Tableau 394. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQUS_NORMAL	0	X'00000000'
TRANSMIS_MQUS_TRANSMISSION	1	X'00000001'

## MQUSAGE\_\* (valeurs d'utilisation de l'ensemble de pages de format de commande et valeurs d'utilisation de l'ensemble de données)

### Valeurs d'utilisation de l'ensemble de pages de format de commande

Tableau 395. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQUSAGE_PS_XX_ENCODE_CASE_ONE disponible	0	X'00000000'
MQUSAGE_PS_DEFINI	1	X'00000001'
MQUSAGE_PS_OFFLINE	2	X'00000002'
MQUSAGE_PS_NOT_DEFINI	3	X'00000003'
MQUSAGE_PS_SUSPENDU	4	X'00000004'
MQUSAGE_EXPAND_USER	1	X'00000001'
MQUSAGE_EXPAND_SYSTEM	2	X'00000002'
MQUSAGE_EXPAND_NONE	3	X'00000003'

### Valeurs d'utilisation du fichier de format de commande

Tableau 396. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQUSAGE_DS_OLDEST_ACTIVE_UOW	10	X'0000000A'
MQUSAGE_DS_OLDEST_PS_RECOVERY	11	X'0000000B'
MQUSAGE_DS_OLDEST_CF_RECOVERY	12	X'0000000C'

## MQVL\_\* (longueur de la valeur)

Tableau 397. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQVL_NULL_TERMINATED	-1	X'FFFFFFFF'
MQVL_EMPTY_CHAINE	0	X'00000000'



## MQVU\_\* (ID utilisateur de variable)

Tableau 398. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQVU_UTILISATEUR_FIXED_USER	1	X'00000001'
Utilisateur MQVU_ANY_USER	2	X'00000002'

## MQWDR\_\* (structure d'enregistrement de destination d'exit de charge de travail de cluster)

Tableau 399. Structures de constantes	
Nom	Structure
ID_STRUC_MQWDR_	"WDR↵"
MQWDR_STRUC_ID_ARRAY	'W', 'D', 'R', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 400. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQWDR_VERSION_1	1	X'00000001'
MQWDR_VERSION_2	2	X'00000002'
MQWDR_VERSION_CURRENT_	2	X'00000002'
MQWDR_LENGTH_1	124	X'0000007C'
MQWDR_LENGTH_2	136	X'00000088'
MQWDR_LONGUEUR_EN_COURS	136	X'00000088'

## MQWI\_\* (intervalle d'attente)

Tableau 401. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQWI_ILLIMITÉ	-1	X'FFFFFFFF'

## MQWIH\_\* (structure d'en-tête d'informations de charge de travail et indicateurs)

### Structure d'en-tête des informations de charge de travail

Tableau 402. Structures de constantes	
Nom	Structure
ID_STRUC_MQWIH	"WIH↵"
MQWIH_STRUC_ID_ARRAY	'W', 'I', 'H', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 403. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQWIH_VERSION_1	1	X'00000001'
MQWIH_CURRENT_VERSION	1	X'00000001'

Tableau 403. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQWIH_LENGTH_1	120	X'00000078'
MQWIH_LONGUEUR_EN_COURS	120	X'00000078'

### Indicateurs d'en-tête d'informations de charge de travail

Tableau 404. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQWIH_AUCUN	0	X'00000000'

### MQWQR\_\* (structure d'enregistrement de file d'attente d'exit de charge de travail de cluster)

Tableau 405. Structures de constantes	
Nom	Structure
MQWQR_STRUC_ID	"WQR~"
MQWQR_STRUC_ID_ARRAY	'W', 'Q', 'R', '~'

**Remarque :** Le symbole ~ représente un caractère blanc unique.

Tableau 406. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQWQR_VERSION_1	1	X'00000001'
MQWQR_VERSION_2	2	X'00000002'
MQWQR_VERSION_3	3	X'00000003'
MQWQR_CURRENT_VERSION	3	X'00000003'
MQWQR_LENGTH_1	200	X'000000C8'
MQWQR_LENGTH_2	208	X'000000D0'
MQWQR_LENGTH_3	212	X'000000D4'
MQWQR_LONGUEUR_EN_COURS	212	X'000000D4'

### MQWS\_\* (Schéma de caractère générique)

Tableau 407. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQWS_VALEUR PAR DEFAULT	0	X'00000000'
MQWS_CHAR	1	X'00000001'
MQWS_TOPIC	2	X'00000002'

### MQWXP\_\* (structure des paramètres d'exit de charge de travail de cluster)

### MQWXP\_\* (structure des paramètres d'exit de charge de travail de cluster)

Tableau 408. Structures de constantes	
Nom	Structure
MQWXP_STRUC_ID	"WXP~"

Tableau 408. Structures de constantes (suite)	
Nom	Structure
MQWXP_STRUC_ID_ARRAY	'W', 'X', 'P', '␣'

**Remarque :** Le symbole ␣ représente un caractère blanc unique.

Tableau 409. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQWXP_VERSION_1	1	X'00000001'
MQWXP_VERSION_2	2	X'00000002'
MQWXP_VERSION_3	3	X'00000003'
MQWXP_VERSION_4	4	X'00000004'
MQWXP_CURRENT_VERSION	4	X'00000004'

### MQWXP\_\* (indicateurs de charge de travail de cluster)

Tableau 410. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQWXP_PUT_BY_CLUSTER_CHL	2	X'00000002'

#### Référence associée

«Zones dans MQWXP -Structure des paramètres d'exit de charge de travail de cluster», à la page 1619  
Description des zones de la structure de paramètres de l'exit de charge de travail de cluster MQWXP

### MQXACT\_\* (types d'appelant d'API)

Tableau 411. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQXACT_EXTERNAL	1	X'00000001'
MQXACT_INTERNAL	2	X'00000002'

### MQXC\_\* (commandes exit)

Tableau 412. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQXC_MQOPEN	1	X'00000001'
MQXC_MQCLOSE	2	X'00000002'
MQXC_MQGET	3	X'00000003'
MQXC_MQPUT	4	X'00000004'
MQXC_MQPUT1	5	X'00000005'
MQXC_MQINQ	6	X'00000006'
MQXC_MQSET	8	X'00000008'
MQXC_MQBACK	9	X'00000009'
MQXC_MQCMIT	10	X'0000000A'

## MQXCC\_\* (réponses d'exit)

Tableau 413. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQXCC_OK	0	X'00000000'
MQXCC_SUPPRESS_FUNCTION	-1	X'FFFFFFFF'
MQXCC_LISTE_FONCTION	-2	X'FFFFFFFE'
MQXCC_SEND_AND_REQUEST_SEC_MSG	-3	X'FFFFFFFD'
MQXCC_SEND_SEC_MSG	-4	X'FFFFFFFC'
MQXCC_SUPPRESS_EXIT	-5	X'FFFFFFFB'
MQXCC_CANAL_FERMÉ	-6	X'FFFFFFFA'
Accusé de réception MQXCC_REQUEST_ACK	-7	X'FFFFFFF9'
MQXCC_ECHEC	-8	X'FFFFFFF8'

## MQXDR\_\* (réponse d'exit)

Tableau 414. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQXDR_OK	0	X'00000000'
Echec de MQXDR_CONVERSION_FAILED	1	X'00000001'

## MQXE\_\* (Environnements)

Tableau 415. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQXE_AUTRES	0	X'00000000'
MQXE_MCA	1	X'00000001'
MQXE_MCA_SVRCONN	2	X'00000002'
SERVEUR_COMMAND_MQ	3	X'00000003'
MQXE_MQSC	4	X'00000004'

## MQXEPO\_\* (structure des options de point d'entrée de registre et options de sortie)

### Structure des options de point d'entrée de registre

Tableau 416. Structures de constantes	
Nom	Structure
MQXEPO_STRUC_ID	"XEPO"
MQXEPO_STRUC_ID_ARRAY	'X', 'E', 'P', 'O'

**Remarque :** Le symbole → représente un caractère blanc unique.

Tableau 417. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQXEPO_VERSION_1	1	X'00000001'
MQXEPO_VERSION_CURRENT_	1	X'00000001'

## Options de sortie

Tableau 418. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQXEPO_NONE	0	X'00000000'

## MQXF\_\* (identificateurs de fonction d'API)

Tableau 419. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQXF_INIT	1	X'00000001'
MQXF_TERME	2	X'00000002'
MQXF_CONN	3	X'00000003'
MQXF_CONNX	4	X'00000004'
MQXF_DISQUE	5	X'00000005'
MQXF_OPEN	6	X'00000006'
MQXF_FERMETURE	7	X'00000007'
MQXF_PUT1	8	X'00000008'
MQXF_PUT	9	X'00000009'
MQXF_GET	10	X'0000000A'
MQXF_DATA_CONV_ON_GET	11	X'0000000B'
MQXF_INQ	12	X'0000000C'
MQXF_SET	13	X'0000000D'
MQXF_DÉBUT	14	X'0000000E'
MQXF_CMIT	15	X'0000000F'
MQXF_RETOUR	16	X'00000010'
MQXF_STAT	18	X'00000012'
MQXF_CB	19	X'00000013'
MQXF_CTL	20	X'00000014'
MQXF_RAPPEL	21	X'00000015'
MQXF_SUB	22	X'00000016'
MQXF_SUBRQ	23	X'00000017'
MQXF_XACLOSE	24	X'00000018'
MQXF_XACOMMIT	25	X'00000019'
MQXF_XATERMINE	26	X'0000001A'
MQXF_XAEND	27	X'0000001B'
MQXF_XAFORGET	28	X'0000001C'
MQXF_XAOPEN	29	X'0000001D'
MQXF_XAPREPARE	30	X'0000001E'
MQXF_XARECOVER	31	X'0000001F'
MQXF_XAROLLBACK	32	X'00000020'
MQXF_XASTART	33	X'00000021'
MQXF_AXREG	34	X'00000022'

Tableau 419. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQXF_AXUNREG	35	X'00000023'

### MQXP\_\* (structure des paramètres d'exit de croisement d'API)

Tableau 420. Structures de constantes	
Nom	Structure
ID_STRUCT_MQXP	"XP↵"
MQXP_STRUC_ID_ARRAY	'X', 'P', '↵', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 421. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQXP_VERSION_1	1	X'00000001'

### MQXPDA\_\* (Zone d'identification des incidents)

Tableau 422. Noms et valeurs de constante	
Nom	Valeur
MQXPDA_NONE	X'00...00' (48 valeurs nulles)
MQXPDA_NONE_ARRAY	'\0', '\0', ... (48 valeurs nulles)

### MQXPT\_\* (types de transport)

Tableau 423. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQXPT_ALL	-1	X'FFFFFFFF'
MQXPT_LOCAL	0	X'00000000'
MQXPT_LU62	1	X'00000001'
MQXPT_TCP	2	X'00000002'
MQXPT_NETBIOS	3	X'00000003'
MQXPT_SPX	4	X'00000004'
MQXPT_DECNET	5	X'00000005'
MQXPT_UDP	6	X'00000006'

### MQXQH\_\* (structure d'en-tête de file d'attente de transmission)

Tableau 424. Structures de constantes	
Nom	Structure
ID_STRUC_MQXQH_	"XQH↵"
MQXQH_TABLE_ID_STRUCTURE	'X', 'Q', 'H', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 425. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQXQH_VERSION_1	1	X'00000001'
MQXQH_CURRENT_VERSION	1	X'00000001'

### MQXR\_\* (Raisons de l'exit)

Tableau 426. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQXR_AVANT	1	X'00000001'
MQXR_APRES	2	X'00000002'
MQXR_CONNECTION	3	X'00000003'
MQXR_INIT	11	X'0000000B'
MQXR_TERM	12	X'0000000C'
MQXR_MSG	13	X'0000000D'
MQXR_XMIT	14	X'0000000E'
MQXR_SEC_MSG	15	X'0000000F'
MQXR_INIT_SEC	16	X'00000010'
MQXR_RETRY	17	X'00000011'
MQXR_AUTO_CLUSSDR	18	X'00000012'
MQXR_AUTO_RECEIVER	19	X'00000013'
MQXR_CLWL_OPEN	20	X'00000014'
MQXR_CLWL_PUT	21	X'00000015'
MQXR_CLWL_MOVE	22	X'00000016'
MQXR_CLWL_REPOS	23	X'00000017'
MQXR_CLWL_REPOS_MOVE	24	X'00000018'
MQXR_END_BATCH	25	X'00000019'
MQXR_ACK_RECEIVED	26	X'0000001A'
MQXR_AUTO_SVRCONN	27	X'0000001B'
MQXR_AUTO_CLUSRCVR	28	X'0000001C'
MQXR_SEC_PARS	29	X'0000001D'

### MQXR2\_\* (réponse d'exit 2)

Tableau 427. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQXR2_PUT_WITH_DEF_ACTION	0	X'00000000'
MQXR2_PUT_WITH_DEF_USERID	1	X'00000001'
MQXR2_PUT_WITH_MSG_USERID	2	X'00000002'
MQXR2_USE_AGENT_BUFFER	0	X'00000000'
MQXR2_USE_EXIT_BUFFER	4	X'00000004'
MQXR2_DEFAULT_CONTINUATION	0	X'00000000'
MQXR2_CONTINUE_CHAIN	8	X'00000008'
MQXR2_SUPPRESS_CHAIN	16	X'00000010'

Tableau 427. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQXR2_STATIC_CACHE	0	X'00000000'
MQXR2_DYNAMIC_CACHE	32	X'00000020'

### MQXT\_\* (identificateurs d'exit)

Tableau 428. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
EXIT MQXT_API_CROSSING_EXIT	1	X'00000001'
MQXT_API_EXIT	2	X'00000002'
MQXT_CHANNEL_SEC_EXIT	11	X'0000000B'
MQXT_CHANNEL_MSG_EXIT	12	X'0000000C'
EXIT MQXT_CHANNEL_SEND_EXIT	13	X'0000000D'
MQXT_CHANNEL_RCV_EXIT	14	X'0000000E'
MQXT_CHANNEL_MSG_RETRY_EXIT	15	X'0000000F'
MQXT_CHANNEL_AUTO_DEF_EXIT	16	X'00000010'
MQXT_CLUSTER_WORKLOAD_EXIT	20	X'00000014'
MQXT_PUBSUB_ROUTING_EXIT	21	X'00000015'

### MQXUA\_\* (valeur de la zone utilisateur d'exit)

Tableau 429. Noms et valeurs de constante	
Nom	Valeur
MQXUA_NONE	X'00...00' (16 valeurs nulles)
MQXUA_NON_ARRAY	'\0', '\0', ... (16 valeurs nulles)

### MQXWD\_\* (structure de descripteur d'attente d'exit)

Tableau 430. Structures de constantes	
Nom	Structure
ID_STRUCD_MQXWD_	"XWD↵"
MQXWD_STRUC_ID_ARRAY	'X', 'W', 'D', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 431. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQXWD_VERSION_1	1	X'00000001'

### MQZAC\_\* (structure de contexte d'application)

Tableau 432. Structures de constantes	
Nom	Structure
ID_STRUC_MQZAC	"ZAC↵"
MQZAC_STRUC_ID_ARRAY	'Z', 'A', 'C', '↵'



**Remarque :** Le symbole – représente un caractère blanc unique.

Tableau 433. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQZAC_VERSION_1	1	X'00000001'
MQZAC_CURRENT_VERSION	1	X'00000001'

### MQZAD\_\* (Structure de données de droits d'accès)

Tableau 434. Structures de constantes

Nom	Structure
ID_STRUC_MQZ	"ZAD–"
MQZAD_STRUC_ID_ARRAY	'Z', 'A', 'D', '–'

**Remarque :** Le symbole – représente un caractère blanc unique.

Tableau 435. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQZAD_VERSION_1	1	X'00000001'
MQZAD_VERSION_2	2	X'00000002'
MQZAD_CURRENT_VERSION	2	X'00000002'

### MQZAET\_\* (types d'entité de services installables)

Tableau 436. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQZAET_AUCUN	0	X'00000000'
MQZAET_PRINCIPAL	1	X'00000001'
GROUPE MQZAET_GROUP	2	X'00000002'
MQZAET_INCONNU	3	X'00000003'

### MQZAO\_\* (Autorisations des services installables)

Tableau 437. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQZAO_CONNECT	1	X'00000001'
MQZAO_PARCOURIR	2	X'00000002'
MQZAO_ENTREE	4	X'00000004'
MQZAO_OUTPUT	8	X'00000008'
MQZAO_INQUIRE	16	X'00000010'
MQZAO_SET	32	X'00000020'
MQZAO_PASS_IDENTITY_CONTEXT	64	X'00000040'
MQZAO_PASS_ALL_CONTEXT	128	X'00000080'
MQZAO_SET_IDENTITY_CONTEXT	256	X'00000100'
MQZAO_SET_ALL_CONTEXT	512	X'00000200'
MQZAO_ALTERNATE_USER_AUTHORITY	1024	X'00000400'
MQZAO_PUBLICATION	2048	X'00000800'

Tableau 437. Valeurs des constantes (suite)		
Nom	Valeur décimale	Valeur hexadécimale
MQZAO_ABONNEMENT	4096	X'00001000'
MQZAO_RESUME	8192	X'00002000'
MQZAO_ALL_MQI	16383	X'00003FFF'
MQZAO_CREER	65536	X'00010000'
MQZAO_DELETE	131072	X'00020000'
MQZAO_DISPLAY	262144	X'00040000'
MODIFICATION MQZAO_DE	524288	X'00080000'
MQZAO_CLEAR	1048576	X'00100000'
CONTROLE MQZ	2097152	X'00200000'
MQZAO_CONTRÔLE_ÉTENDU	4194304	X'00400000'
MQZAO_AUTORISATION	8388608	X'00800000'
MQZAO_ALL_ADMIN	16646144	X'00FE0000'
MQZAO_ALL	16662527	X'00FE3FFF'
MQZAO_RETRAIT	16777216	X'01000000'
MQZAO_AUCUN	0	X'00000000'

### MQZAS\_\* (version de l'interface de service des services installables)

Tableau 438. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQZAS_VERSION_1	1	X'00000001'
MQZAS_VERSION_2	2	X'00000002'
MQZAS_VERSION_3	3	X'00000003'
MQZAS_VERSION_4	4	X'00000004'
MQZAS_VERSION_5	5	X'00000005'
MQZAS_VERSION_6	6	X'00000006'

### MQZAT\_\* (types d'authentification)

Tableau 439. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQZAT_INITIAL_CONTEXT	0	X'00000000'
MQZAT_CHANGE_CONTEXT	1	X'00000001'

### MQZCI\_\* (indicateur de continuation des services installables)

Tableau 440. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQZCI_DEFAULT	0	X'00000000'
MQZCI_CONTINUER	0	X'00000000'
MQZCI_ARRET	1	X'00000001'

## MQZED\_\* (structure de données d'entité)

Tableau 441. Structures de constantes	
Nom	Structure
ID_STRUC_MQZ	"ZED↵"
MQZED_STRUC_ID_ARRAY	'Z', 'E', 'D', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 442. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQZED_VERSION_1	1	X'00000001'
MQZED_VERSION_2	2	X'00000002'
VERSION_MQZED_CURRENT_VERSION	2	X'00000002'

## MQZFP\_\* (structure des paramètres disponibles)

Tableau 443. Structures de constantes	
Nom	Structure
ID_STRUC_MQZFP	"ZFP↵"
MQZFP_STRUC_ID_ARRAY	'Z', 'F', 'P', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 444. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQZFP_VERSION_1	1	X'00000001'
VERSION_MQZFP_CURRENT_VERSION	1	X'00000001'

## MQZIC\_\* (structure de contexte d'identité)

Tableau 445. Structures de constantes	
Nom	Structure
ID_STRUC_MQZ	"ZIC↵"
MQZIC_STRUC_ID_ARRAY	'Z', 'I', 'C', '↵'

**Remarque :** Le symbole ↵ représente un caractère blanc unique.

Tableau 446. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQZIC_VERSION_1	1	X'00000001'
MQZIC_CURRENT_VERSION	1	X'00000001'

## MQZID\_ \* (ID de fonction pour les services)

### ID de fonction communs à tous les services

Tableau 447. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQZID_INIT	0	X'00000000'
MQZID_TERM	1	X'00000001'

### ID de fonction pour le service de droits d'accès

Tableau 448. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQZID_INIT_AUTHORITY	0	X'00000000'
MQZID_TERM_AUTHORITY	1	X'00000001'
MQZID_CHECK_AUTHORITY	2	X'00000002'
MQZID_COPY_ALL_AUTHORITY	3	X'00000003'
MQZID_DELETE_AUTHORITY	4	X'00000004'
MQZID_SET_AUTHORITY	5	X'00000005'
MQZID_GET_AUTHORITY	6	X'00000006'
MQZID_GET_EXPLICIT_AUTHORITY	7	X'00000007'
MQZID_REFRESH_CACHE	8	X'00000008'
MQZID_ENUMERATE_AUTHORITY_DATA	9	X'00000009'
MQZID_AUTHENTICATE_USER	10	X'0000000A'
Utilisateur_FREE_MQZ	11	X'0000000B'
MQZID_INQUIRE	12	X'0000000C'
MQZID_CHECK_PRIVILEGED	13	X'0000000D'

### ID de fonction pour le service Name

Tableau 449. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
NOM_INIT_MQZID	0	X'00000000'
NOM_MQZID_TERM	1	X'00000001'
MQZID_LOOKUP_NAME	2	X'00000002'
MQZID_INSERT_NAME	3	X'00000003'
NOM_DELETE_MQZ	4	X'00000004'

### ID de fonction pour le service Userid

Tableau 450. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
ID_INIT_ID_MQZID	0	X'00000000'
ID utilisateur MQZID_TERM_	1	X'00000001'
MQZID_FIND_USERID	2	X'00000002'

## MQZIO\_\* (Options d'initialisation des services installables)

Tableau 451. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQZIO_PRIMAIRE	0	X'00000000'
MQZIO_SECONDAIRE	1	X'00000001'

## MQZNS\_\* (Version de l'interface de service de nom)

Tableau 452. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQZNS_VERSION_1	1	X'00000001'

## MQZSE\_\* (Indicateur de début-énumération des services installables)

Tableau 453. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
DEMARRAGE MQZ	1	X'00000001'
MQZSE_CONTINUE	0	X'00000000'

## MQZSL\_\* (indicateur de sélecteur de services installables)

Tableau 454. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQZSL_NOT_RETURNED	0	X'00000000'
MQZSL_RENVOYE	1	X'00000001'

## MQZTO\_\* (Options de résiliation des services installables)

Tableau 455. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQZTO_PRIMAIRE	0	X'00000000'
MQZTO_SECONDAIRE	1	X'00000001'

## MQZUS\_\* (version de l'interface de service d'ID utilisateur)

Tableau 456. Valeurs des constantes		
Nom	Valeur décimale	Valeur hexadécimale
MQZUS_VERSION_1	1	X'00000001'

## Types de données utilisés dans l'interface MQI

Informations sur les types de données pouvant être utilisés dans l'interface MQI (Message Queue Interface). Descriptions, zones et déclarations de langue pour les langues appropriées avec chaque type de données.

## Types de données et programmation pour l'interface MQI

Présentation des types de données Elementary et Structure et comment utiliser l'interface MQI via la programmation C, la programmation COBOL ou la programmation High Level Assembler .

## **Types de données élémentaires**

Informations sur les types de données utilisés dans l'interface MQI ou dans les fonctions d'exit. Ils sont décrits en détail, suivis d'exemples montrant comment déclarer les types de données élémentaires dans les langages de programmation pris en charge.

Les types de données utilisés dans l'interface MQI ou dans les fonctions d'exit sont les suivants:

- Types de données élémentaires, ou
- Agrégats de types de données élémentaires (tableaux ou structures)

Les types de données élémentaires suivants sont utilisés dans l'interface MQI ou dans les fonctions d'exit:

<i>Tableau 457. Noms, types et descriptions des types de données élémentaires</i>		
<b>Nom du type de données élémentaires</b>	<b>Type de données</b>	<b>Description</b>
MQBOOL	Boolean	Le type de données MQBOOL représente une valeur booléenne. La valeur 0 représente false. Toute autre valeur représente true. Un MQBOOL doit être aligné comme pour le type de données MQLONG.

Tableau 457. Noms, types et descriptions des types de données élémentaires (suite)

Nom du type de données élémentaires	Type de données	Description
MQBYTE	Octet	<p>Le type de données MQBYTE représente un octet de données unique. Aucune interprétation particulière n'est placée sur l'octet ; elle est traitée comme une chaîne de bits, et non comme un nombre ou un caractère binaire. Aucun alignement spécial n'est requis.</p> <p>Lorsque des données MQBYTE sont envoyées entre des gestionnaires de files d'attente qui utilisent des jeux de caractères ou des codages différents, les données MQBYTE ne sont converties d'aucune manière. Les zones <i>MsgId</i> et <i>CorrelId</i> de la structure MQMD sont similaires à ceci.</p> <p>Un tableau de MQBYTE est parfois utilisé pour représenter une zone de la mémoire principale qui n'est pas connue du gestionnaire de files d'attente. Par exemple, la zone peut contenir des données de message d'application ou une structure. L'alignement des limites de cette zone doit être compatible avec la nature des données qu'elle contient.</p> <p>Dans le langage de programmation C, tout type de données peut être utilisé pour les paramètres de fonction affichés sous forme de tableaux de MQBYTE. En effet, ces paramètres sont toujours transmis par adresse, et en C, le paramètre de fonction est déclaré comme un pointeur vers vide.</p>

Tableau 457. Noms, types et descriptions des types de données élémentaires (suite)

Nom du type de données élémentaires	Type de données	Description
MQBYTE $n$	Chaîne de $n$ octets	<p>Chaque type de données MQBYTE<math>n</math> représente une chaîne de <math>n</math> octets, où <math>n</math> peut prendre l'une des valeurs suivantes: 8, 16, 24, 32, 40 ou 128. Chaque octet est décrit par le type de données MQBYTE. Aucun alignement spécial n'est requis.</p> <p>Si les données de la chaîne d'octets sont plus courtes que la longueur définie de la chaîne, les données doivent être complétées par des valeurs nulles pour remplir la chaîne.</p> <p>Lorsque le gestionnaire de files d'attente renvoie des chaînes d'octets à l'application (par exemple, sur l'appel MQGET), le gestionnaire de files d'attente remplit avec des valeurs nulles la longueur définie de la chaîne.</p> <p>Les constantes nommées sont disponibles pour définir les longueurs des zones de chaîne d'octets. Ces éléments sont répertoriés dans le «Constantes», à <a href="#">la page 61</a></p>



Tableau 457. Noms, types et descriptions des types de données élémentaires (suite)

Nom du type de données élémentaires	Type de données	Description
MQCHAR	Caractère	<p>Le type de données MQCHAR représente un caractère à un octet ou un octet d'un caractère à deux octets ou à plusieurs octets. Aucun alignement spécial n'est requis.</p> <p>Lorsque des données MQCHAR sont envoyées entre des gestionnaires de files d'attente qui utilisent des jeux de caractères ou des codages différents, les données MQCHAR doivent généralement être converties pour que les données soient interprétées correctement. Le gestionnaire de files d'attente effectue cette opération automatiquement pour les données MQCHAR dans la structure MQMD. La conversion des données MQCHAR dans les données de message d'application est contrôlée par l'option MQGMO_CONVERT spécifiée dans l'appel MQGET ; voir la description de cette option dans «Options MQGMO-Get-message», à la page <a href="#">382</a> pour plus de détails.</p>

Tableau 457. Noms, types et descriptions des types de données élémentaires (suite)

Nom du type de données élémentaires	Type de données	Description
MQCHARn	Chaîne de $n$ caractères	<p>Chaque type de données MQCHARn représente une chaîne de <math>n</math> caractères, où <math>n</math> peut prendre l'une des valeurs suivantes: 4, 8, 12, 20, 28, 32, 48, 64, 128 ou 256. Chaque caractère est décrit par le type de données MQCHAR. Aucun alignement spécial n'est requis.</p> <p>Si les données de la chaîne sont plus courtes que la longueur définie de la chaîne, les données doivent être complétées par des blancs pour remplir la chaîne. Dans certains cas, un caractère nul peut être utilisé pour terminer la chaîne prématurément, au lieu d'être rempli avec des blancs ; le caractère nul et les caractères qui le suivent sont traités comme des blancs, jusqu'à la longueur définie de la chaîne. Les emplacements où une valeur nulle peut être utilisée sont identifiés dans les descriptions d'appel et de type de données.</p> <p>Lorsque le gestionnaire de files d'attente renvoie des chaînes de caractères à l'application (par exemple, sur l'appel MQGET), le gestionnaire de files d'attente remplit toujours avec des blancs la longueur définie de la chaîne ; le gestionnaire de files d'attente n'utilise pas le caractère null pour délimiter la chaîne.</p> <p>Des constantes nommées sont disponibles pour définir les longueurs des zones de chaîne de caractères et sont répertoriées dans «Constantes», à la page 61.</p>

Tableau 457. Noms, types et descriptions des types de données élémentaires (suite)

Nom du type de données élémentaires	Type de données	Description
MQFLOAT32	Nombre à virgule flottante 32 bits	<p>Le type de données MQFLOAT32 est un nombre à virgule flottante 32 bits représenté à l'aide du format à virgule flottante IEEE standard. Un MQFLOAT32 doit être aligné sur une limite de 4 octets.</p> <p>L'utilisation de MQFLOAT32 dans C sous z/OS nécessite l'utilisation de l'indicateur de compilateur FLOAT (IEEE).</p> <p>L'utilisation de MQFLOAT32 en COBOL est limitée aux compilateurs qui prennent en charge les nombres à virgule flottante au format IEEE. Cela peut nécessiter l'utilisation de l'indicateur de compilateur FLOAT (NATIVE).</p>
MQFLOAT64	Nombre à virgule flottante 64 bits	<p>Le type de données MQFLOAT64 est un nombre à virgule flottante 64 bits représenté à l'aide du format à virgule flottante IEEE standard. Un MQFLOAT64 doit être aligné sur une limite de 8 octets.</p> <p>L'utilisation de MQFLOAT64 en C sous z/OS nécessite l'utilisation de l'indicateur de compilateur FLOAT (IEEE).</p> <p>L'utilisation de MQFLOAT64 en COBOL est limitée aux compilateurs qui prennent en charge les nombres à virgule flottante au format IEEE. Cela peut nécessiter l'utilisation de l'indicateur de compilateur FLOAT (NATIVE).</p>

Tableau 457. Noms, types et descriptions des types de données élémentaires (suite)

Nom du type de données élémentaires	Type de données	Description
MQHCONFIG	Descripteur de configuration	<p>Le type de données MQHCONFIG représente un descripteur de configuration, c'est-à-dire le composant configuré pour un service installable particulier. Un descripteur de configuration doit être aligné sur sa limite naturelle.</p> <p>Les applications ne doivent pas s'appuyer sur le format des données stockées dans ce descripteur. Si elle est valide, sa valeur est destinée à être utilisable dans d'autres appels MQI, mais n'est pas destinée à avoir de signification en dehors de cet objectif.</p>
MQHCONN	Descripteur de connexion	<p>Le type de données MQHCONN représente un descripteur de connexion, c'est-à-dire la connexion à un gestionnaire de files d'attente particulier. Un descripteur de connexion doit être aligné sur une limite de 4 octets.</p> <p>Les applications ne doivent pas s'appuyer sur le format des données stockées dans ce descripteur. Si elle est valide, sa valeur est destinée à être utilisable dans d'autres appels MQI, mais n'est pas destinée à avoir de signification en dehors de cet objectif.</p>
MQHMSG	descripteur de message	<p>Le type de données MQHMSG représente un descripteur de message qui donne accès à un message. Un descripteur de message doit être aligné sur une limite de 8 octets.</p> <p>Les applications ne doivent pas s'appuyer sur le format des données stockées dans ce descripteur. Si elle est valide, sa valeur est destinée à être utilisable dans d'autres appels MQI, mais n'est pas destinée à avoir de signification en dehors de cet objectif.</p>

Tableau 457. Noms, types et descriptions des types de données élémentaires (suite)

Nom du type de données élémentaires	Type de données	Description
MQHOBJ	Descripteur d'objet	<p>Le type de données MQHOBJ représente un descripteur d'objet qui donne accès à un objet. Un descripteur d'objet doit être aligné sur une limite de 4 octets.</p> <p>Les applications ne doivent pas s'appuyer sur le format des données stockées dans ce descripteur. Si elle est valide, sa valeur est destinée à être utilisable dans d'autres appels MQI, mais n'est pas destinée à avoir de signification en dehors de cet objectif.</p>
MQINT8	Entier signé 8 bits	Le type de données MQINT8 est un entier signé de 8 bits qui peut prendre n'importe quelle valeur comprise entre -128 et +127, sauf indication contraire du contexte.
MQINT16	Entier signé 16 bits	Le type de données MQINT16 est un entier signé 16 bits qui peut prendre n'importe quelle valeur comprise entre -32 768 et +32 767, sauf indication contraire du contexte. Un MQINT16 doit être aligné sur une limite de 2 octets.
MQINT32	Entier signé 32 bits	<p>Le type de données MQINT32 est un entier binaire signé 32 bits qui peut prendre n'importe quelle valeur comprise entre -2 147 483 648 et + 2 147 483 647, sauf restriction par le contexte.</p> <p>Voir la définition de <u>MQLONG</u>.</p>
MQINT64	Entier signé sur 64 bits	<p>Le type de données MQINT64 est un entier signé de 64 bits qui peut prendre n'importe quelle valeur comprise entre -9 223 372 036 854 775 808 et + 9 223 372 036 854 775 807, sauf restriction par le contexte.</p> <p>Pour COBOL, la plage valide est limitée à -999 999 999 999 999 999 999 à +999 999 999 999 999 999. Un entier 64 bits doit être aligné sur une limite de 8 octets.</p>

Tableau 457. Noms, types et descriptions des types de données élémentaires (suite)

Nom du type de données élémentaires	Type de données	Description
MQLONG	Entier signé 32 bits	<p>Le type de données MQLONG est un entier binaire signé 32 bits qui peut prendre n'importe quelle valeur comprise entre -2 147 483 648 et + 2 147 483 647, sauf restriction par le contexte.</p> <p>Pour COBOL, la plage valide est comprise entre -999 999 999 et +999 999 999. Un MQLONG doit être aligné sur une limite de 4 octets.</p>
MQPID	Identificateur de processus	<p>Identificateur de processus IBM MQ .</p> <p>Il s'agit du même identificateur utilisé dans la trace MQ et les vidages FFST™, mais il peut être différent de l'identificateur de processus du système d'exploitation.</p>
MQPTR	Pointeur	<p>Le type de données MQPTR est l'adresse des données de n'importe quel type. Un pointeur doit être aligné sur sa limite naturelle ; il s'agit d'une limite de 16 octets sur IBM i et d'une limite de 8 octets sur d'autres plateformes.</p> <p>Certains langages de programmation prennent en charge les pointeurs typés ; l'interface MQI les utilise également dans quelques cas (par exemple, PMQCHAR et PMQLONG dans le langage de programmation C).</p>
MQTID	Identificateur d'unité d'exécution	<p>Identificateur de l'unité d'exécution IBM MQ .</p> <p>Il s'agit du même identificateur utilisé dans la trace MQ et les vidages FFST™, mais il peut être différent de l'identificateur d'unité d'exécution du système d'exploitation.</p>

Tableau 457. Noms, types et descriptions des types de données élémentaires (suite)

Nom du type de données élémentaires	Type de données	Description
MQUINT8	Entier non signé 8 bits	Le type de données MQUINT8 est un entier non signé de 8 bits qui peut prendre n'importe quelle valeur comprise entre 0 et +255, sauf indication contraire du contexte.
MQUINT16	Entier non signé 16 bits	Le type de données MQUINT16 est un entier non signé de 16 bits qui peut prendre n'importe quelle valeur comprise entre 0 et +65 535, sauf restriction par le contexte. Une MQUINT16 doit être alignée sur une limite de 2 octets.
MQUINT32	entier non marqué 32-bit	Le type de données MQUINT32 est un entier binaire non signé 32 bits. Voir la définition de <a href="#">MQULONG</a> .
MQUINT64	Entier non signé sur 64 bits	Le type de données MQUINT64 est un entier non signé de 64 bits qui peut prendre n'importe quelle valeur comprise entre 0 et +18 446 744 073 709 551 615, sauf restriction par le contexte. Pour COBOL, la plage valide est comprise entre 0 et +999 999 999 999 999 999. Un entier 64 bits doit être aligné sur une limite de 8 octets.
MQULONG	entier non marqué 32-bit	Le type de données MQULONG est un entier binaire non signé de 32 bits qui peut prendre n'importe quelle valeur comprise entre 0 et + 4 294 967 294, sauf restriction par le contexte. Pour COBOL, la plage valide est comprise entre 0 et +999 999 999. Une valeur MQULONG doit être alignée sur une limite de 4 octets.
PMQACH	Pointeur	Pointeur vers une structure de données de type MQACH
PMQAIR	Pointeur	Pointeur vers une structure de données de type MQAIR
PMQAXC	Pointeur	Pointeur vers une structure de données de type MQAXC
PMQAXP	Pointeur	Pointeur vers une structure de données de type MQAXP

Tableau 457. Noms, types et descriptions des types de données élémentaires (suite)

<b>Nom du type de données élémentaires</b>	<b>Type de données</b>	<b>Description</b>
PMQBMHO	Pointeur	Pointeur vers une structure de données de type MQBMHO
PMQBO	Pointeur	Pointeur vers une structure de données de type MQBO
PMQBOOL	Pointeur	Pointeur vers des données de type MQBOOL
PMQBYTE	Pointeur	Pointeur vers des données de type MQBYTE
PMQBYTEn	Pointeur	Pointeur vers des données de type MQBYTEn, où n peut être 8, 16, 24, 32, 40, 128
PMQCBC	Pointeur	Pointeur vers une structure de données de type MQCBC
PMQCBD	Pointeur	Pointeur vers une structure de données de type MQCBD
PMQCHAR	Pointeur	Pointeur vers des données de type MQCHAR
PMQCHARN	Pointeur	Pointeur vers un type de données de MQCHARN, où n peut être 4, 8, 12, 20, 28, 32, 48, 64, 128, 256, 264
PMQCHARV	Pointeur	Pointeur vers une structure de données de type MQCHARV
PMQCIH	Pointeur	Pointeur vers une structure de données de type MQCIH
PMQCMHO	Pointeur	Pointeur vers une structure de données de type MQCMHO
PMQCNO	Pointeur	Pointeur vers une structure de données de type MQCNO
PMQCSP	Pointeur	Pointeur vers une structure de données de type MQCSP
PMQCTLO	Pointeur	Pointeur vers une structure de données de type MQCTLO
PMQDH	Pointeur	Pointeur vers une structure de données de type MQDH
PMQDHO	Pointeur	Pointeur vers une structure de données de type MQDHO
PMQDLH	Pointeur	Pointeur vers une structure de données de type MQDLH
PMQDMHO	Pointeur	Pointeur vers une structure de données de type MQDMHO



Tableau 457. Noms, types et descriptions des types de données élémentaires (suite)

<b>Nom du type de données élémentaires</b>	<b>Type de données</b>	<b>Description</b>
PMQDMPO	Pointeur	Pointeur vers une structure de données de type MQDMPO
PMQEPH	Pointeur	Pointeur vers une structure de données de type MQEPH
PMQFLOAT32	Pointeur	Pointeur vers une structure de données de type MQFLOAT32
PMQFLOAT64	Pointeur	Pointeur vers une structure de données de type MQFLOAT64
PMQFUNC	Pointeur	Pointeur vers une fonction
PMQGMPO	Pointeur	Pointeur vers une structure de données de type MQGMPO
PMQHCONFIG	Pointeur	Pointeur vers des données de type MQHCONFIG
PMQHCONN	Pointeur	Pointeur vers des données de type MQHCONN
PMQHMSG	Pointeur	Pointeur vers des données de type MQHMSG
PMQHOBJ	Pointeur	Pointeur vers des données de type MQHOBJ
PMQIIH	Pointeur	Pointeur vers une structure de données de type MQIIH
PMQIMPO	Pointeur	Pointeur vers une structure de données de type MQIMPO
PMQINT8	Pointeur	Pointeur vers des données de type MQINT8
PMQINT16	Pointeur	Pointeur vers des données de type MQINT16
PMQINT32	Pointeur	Pointeur vers des données de type MQINT32
PMQINT64	Pointeur	Pointeur vers des données de type MQINT64
PMQLONG	Pointeur	Pointeur vers des données de type MQLONG
PMQMD	Pointeur	Pointeur vers la structure de type MQMD
PMQMDE	Pointeur	Pointeur vers une structure de données de type MQMDE
PMQMD1	Pointeur	Pointeur vers une structure de données de type MQMD1
PMQMD2	Pointeur	Pointeur vers une structure de données de type MQMD2

Tableau 457. Noms, types et descriptions des types de données élémentaires (suite)

<b>Nom du type de données élémentaires</b>	<b>Type de données</b>	<b>Description</b>
PMQMHBO	Pointeur	Pointeur vers une structure de données de type MQMHBO
PMQOD	Pointeur	Pointeur vers une structure de données de type MQOD
PMQOR	Pointeur	Pointeur vers une structure de données de type MQOR
PMQPD	Pointeur	Pointeur vers une structure de données de type MQPD
PMQPID	Pointeur	Pointeur vers un identificateur de processus
PMQMD	Pointeur	Pointeur vers une structure de données de type MQMD
PMQPMO	Pointeur	Pointeur vers une structure de données de type MQPMO
PMQPTR	Pointeur	Pointeur vers des données de type MQPTR
PMQRFH	Pointeur	Pointeur vers une structure de données de type MQRFH
PMQRFH2	Pointeur	Pointeur vers une structure de données de type MQRFH2
PMQRMH	Pointeur	Pointeur vers une structure de données de type MQRMH
PMQRR	Pointeur	Pointeur vers une structure de données de type MQRR
PMQSCO	Pointeur	Pointeur vers une structure de données de type MQSCO
PMQSD	Pointeur	Pointeur vers une structure de données de type MQSD
PMQSMPO	Pointeur	Pointeur vers une structure de données de type MQSMPO
PMQSRO	Pointeur	Pointeur vers une structure de données de type MQSRO
PMSSTS	Pointeur	Pointeur vers une structure de données de type MQSTS
PMQTID	Pointeur	Pointeur vers un ID d'unité d'exécution
PMQTM	Pointeur	Pointeur vers une structure de données de type MQTM
PMQTM2	Pointeur	Pointeur vers une structure de données de type MQTM2

Tableau 457. Noms, types et descriptions des types de données élémentaires (suite)

Nom du type de données élémentaires	Type de données	Description
PMQUINT8	Pointeur	Pointeur vers un type de données MQUINT8
PMQUINT16	Pointeur	Pointeur vers un type de données de MQUINT16
PMQUINT32	Pointeur	Pointeur vers un type de données de MQUINT32
PMQUINT64	Pointeur	Pointeur vers un type de données de MQUINT64
PPMQULONG	Pointeur	Pointeur vers un type de données de MQULONG
ID PMQVOID	Pointeur	
PMQWIH	Pointeur	Pointeur vers une structure de données de type MQWIH
PMQXQH	Pointeur	Pointeur vers une structure de données de type MQXQH

Déclarations de type de données C

Tableau 458. Noms et représentations des types de données C

Type de données	Représentation
MQBOOL	<pre>typedef MQLONG MQBOOL;</pre>
MQBYTE	<pre>typedef unsigned char MQBYTE;</pre>
MQBYTE8	<pre>typedef MQBYTE MQBYTE8[8];</pre>
MQBYTE16	<pre>typedef MQBYTE MQBYTE16[16];</pre>
MQBYTE24	<pre>typedef MQBYTE MQBYTE24[24];</pre>
MQBYTE32	<pre>typedef MQBYTE MQBYTE32[32];</pre>
MQBYTE40	<pre>typedef MQBYTE MQBYTE40[40];</pre>
MQCHAR	<pre>typedef char MQCHAR;</pre>

Tableau 458. Noms et représentations des types de données C (suite)

Type de données	Représentation
MQCHAR4	<code>typedef MQCHAR MQCHAR4[4];</code>
MQCHAR8	<code>typedef MQCHAR MQCHAR8[8];</code>
MQCHAR12	<code>typedef MQCHAR MQCHAR12[12];</code>
MQCHAR20	<code>typedef MQCHAR MQCHAR20[20];</code>
MQCHAR28	<code>typedef MQCHAR MQCHAR28[28];</code>
MQCHAR32	<code>typedef MQCHAR MQCHAR32[32];</code>
MQCHAR48	<code>typedef MQCHAR MQCHAR48[48];</code>
MQCHAR64	<code>typedef MQCHAR MQCHAR64[64];</code>
MQCHAR128	<code>typedef MQCHAR MQCHAR128[128];</code>
MQCHAR256	<code>typedef MQCHAR MQCHAR256[256];</code>
MQFLOAT32	<code>typedef float MQFLOAT32;</code>
MQFLOAT64	<code>typedef double MQFLOAT64;</code>
MQHCONFIG	<code>typedef void MQPOINTER MQHCONFIG;</code>
MQHCONN	<code>typedef MQLONG MQHCONN;</code>
MQHOBJ	<code>typedef MQLONG MQHOBJ;</code>
MQINT8	<code>typedef signed char MQINT8;</code>

Tableau 458. Noms et représentations des types de données C (suite)

Type de données	Représentation
MQINT16	<pre>typedef short MQINT16;</pre>
MQINT64	<p><b>UNIX</b> Sous UNIX64 bits:</p> <pre>typedef long;</pre> <p><b>AIX</b> Sous AIX:</p> <pre>typedef int64_t;</pre> <p><b>IBM i</b> <b>Linux</b> <b>z/OS</b> Sous Linux, IBM i et z/OS:</p> <pre>typedef long long;</pre> <p><b>Windows</b> Sous Windows :</p> <pre>typedef _int64;</pre>
MQLONG	<p><b>IBM i</b> Sous IBM i :</p> <pre>typedef long MQLONG;</pre> <p><b>z/OS</b> <b>ALW</b> Sur d'autres plateformes:</p> <pre>if defined(MQ_64_BIT)     typedef int MQLONG; else     typedef long MQLONG;</pre>
MQPID	<pre>typedef MQLONG MQPID;</pre>
MQPTR	<pre>typedef void MQPOINTER MQPTR;</pre>
MQTID	<pre>typedef MQLONG MQTID;</pre>
MQUINT8	<pre>typedef unsigned char MQUINT8;</pre>
MQUINT16	<pre>typedef unsigned short MQUINT16;</pre>

Tableau 458. Noms et représentations des types de données C (suite)

Type de données	Représentation
MQUINT64	<p><b>UNIX</b> Sous UNIX64 bits:</p> <pre>typedef unsigned long;</pre> <p><b>AIX</b> Sous AIX:</p> <pre>typedef uint64_t;</pre> <p><b>IBM i</b> <b>Linux</b> <b>z/OS</b> Sous Linux, IBM i et z/OS:</p> <pre>typedef unsigned long long;</pre> <p><b>Windows</b> Sous Windows :</p> <pre>typedef unsigned _int64;</pre>
MQULONG	<p><b>IBM i</b> Sous IBM i :</p> <pre>typedef unsigned long MQULONG;</pre> <p><b>z/OS</b> <b>ALW</b> Sur d'autres plateformes:</p> <pre>if defined(MQ_64_BIT)     typedef unsigned int MQULONG; else     typedef unsigned long MQULONG;</pre>
PMQBO	<pre>typedef MQBO MQPOINTER PMQBO;</pre>
PMQBOOL	<pre>typedef MQBOOL MQPOINTER PMQBOOL;</pre>
PMQBYTE	<pre>typedef MQBYTE MQPOINTER PMQBYTE;</pre>
PMQBYTE8	<pre>typedef MQBYTE8[8] MQPOINTER PMQBYTE8[8];</pre>
PMQBYTE16	<pre>typedef MQBYTE16[16] MQPOINTER PMQBYTE16[16];</pre>
PMQBYTE24	<pre>typedef MQBYTE24[24] MQPOINTER PMQBYTE24[24];</pre>

Tableau 458. Noms et représentations des types de données C (suite)

Type de données	Représentation
PMQBYTE32	<code>typedef MQBYTE32[32] MQPOINTER PMQBYTE32[32];</code>
PMQBYTE40	<code>typedef MQBYTE40[40] MQPOINTER PMQBYTE40[40];</code>
PMQBYTE128	<code>typedef MQBYTE128[128] MQPOINTER PMQBYTE128[128];</code>
PMQCHAR	<code>typedef MQCHAR MQPOINTER PMQCHAR;</code>
PMQCHAR4	<code>typedef MQCHAR4[4] MQPOINTER PMQCHAR4[4];</code>
PMQCHAR8	<code>typedef MQCHAR8[8] MQPOINTER PMQCHAR8[8];</code>
PMQCHAR12	<code>typedef MQCHAR12[12] MQPOINTER PMQCHAR12[12];</code>
PMQCHAR20	<code>typedef MQCHAR20[20] MQPOINTER PMQCHAR20[20];</code>
PMQCHAR28	<code>typedef MQCHAR28[28] MQPOINTER PMQCHAR28[28];</code>
PMQCHAR32	<code>typedef MQCHAR32[32] MQPOINTER PMQCHAR32[32];</code>
PMQCHAR48	<code>typedef MQCHAR48[48] MQPOINTER PMQCHAR48[48];</code>
PMQCHAR64	<code>typedef MQCHAR64[64] MQPOINTER PMQCHAR64[64];</code>
PMQCHAR128	<code>typedef MQCHAR128[128] MQPOINTER PMQCHAR128[128];</code>
PMQCHAR256	<code>typedef MQCHAR256[256] MQPOINTER PMQCHAR256[256];</code>
PMQCHAR264	<code>typedef MQCHAR264[264] MQPOINTER PMQCHAR264[264];</code>
PMQCIH	<code>typedef MQCIH MQPOINTER PMQCIH;</code>

Tableau 458. Noms et représentations des types de données C (suite)

Type de données	Représentation
PMQCNO	<code>typedef MQCNO MQPOINTER PMQCNO;</code>
PMQDLH	<code>typedef MQDLH MQPOINTER PMQDLH;</code>
PMQFUNC	<code>typedef void MQPOINTER PMQFUNC;</code>
PMQFLOAT32	<code>typedef MQFLOAT32 MQPOINTER PMQFLOAT32;</code>
PMQFLOAT64	<code>typedef MQFLOAT64 MQPOINTER PMQFLOAT64;</code>
PMQGMO	<code>typedef MQGMO MQPOINTER PMQGMO;</code>
PMQHCONFIG	<code>typedef MQHCONFIG MQPOINTER PMQHCONFIG;</code>
PMQHCONN	<code>typedef MQHCONN MQPOINTER PMQHCONN;</code>
PMQHOBJ	<code>typedef MQHOBJ MQPOINTER PMQHOBJ;</code>
PMQIIH	<code>typedef MQIIH MQPOINTER PMQIIH;</code>
PMQINT8	<code>typedef MQINT8 MQPOINTER PMQINT8;</code>
PMQINT16	<code>typedef MQINT16 MQPOINTER PMQINT16;</code>
PMQLONG	<code>typedef MQLONG MQPOINTER PMQLONG;</code>
PMQMD	<code>typedef MQMD MQPOINTER PMQMD;</code>
PMQMD1	<code>typedef MQMD1[1] MQPOINTER PMQMD1[1];</code>
PMQMDE	<code>typedef MQMDE MQPOINTER PMQMDE;</code>



Tableau 458. Noms et représentations des types de données C (suite)

Type de données	Représentation
PMQOD	<code>typedef MQOD MQPOINTER PMQOD;</code>
PMQPMO	<code>typedef MQPMO MQPOINTER PMQPMO;</code>
PMQPTR	<code>typedef MQPTR MQPOINTER PMQPTR;</code>
PMQRFH	<code>typedef MQRFH MQPOINTER PMQRFH;</code>
PMQRFH2	<code>typedef MQRFH2[2] MQPOINTER PMQRFH2[2];</code>
PMQRMH	<code>typedef MQRMH MQPOINTER PMQRMH;</code>
PMQTM	<code>typedef MQTM MQPOINTER PMQTM;</code>
PMQTM2	<code>typedef MQTM2[2] MQPOINTER PMQTM2[2];</code>
PMQUINT8	<code>typedef MQUINT8 MQPOINTER PMQUINT8;</code>
PMQUINT16	<code>typedef MQUINT16 MQPOINTER PMQUINT16;</code>
PPMQULONG	<code>typedef MQULONG MQPOINTER PPMQULONG;</code>
ID PMQVOID	<code>typedef void MQPOINTER PMQVOID;</code>
PMQWIH	<code>typedef MQWIH MQPOINTER PMQWIH;</code>
PMQXQH	<code>typedef MQXQH MQPOINTER PMQXQH;</code>
PPMQBO	<code>typedef PMQBO MQPOINTER PPMQBO;</code>
PPMQBYTE (octet)	<code>typedef PMQBYTE MQPOINTER PPMQBYTE;</code>

Tableau 458. Noms et représentations des types de données C (suite)

Type de données	Représentation
PPMQCHAR	<code>typedef PMQCHAR MQPOINTER PPMQCHAR;</code>
PPMQCNO	<code>typedef PMQCNO MQPOINTER PPMQCNO;</code>
PPMQGMO	<code>typedef PMQGMO MQPOINTER PPMQGMO;</code>
PPMQHCONN	<code>typedef PMQHCONN MQPOINTER PPMQHCONN;</code>
PPMQHOBJ	<code>typedef PMQHOBJ MQPOINTER PPMQHOBJ;</code>
PPMQLONG	<code>typedef PMQLONG MQPOINTER PPMQLONG;</code>
PPMQMD	<code>typedef PMQMD MQPOINTER PPMQMD;</code>
PPMQOD	<code>typedef PMQOD MQPOINTER PPMQOD;</code>
PPMQPMO	<code>typedef PMQPMO MQPOINTER PPMQPMO;</code>
PPMQULONG	<code>typedef PMQULONG MQPOINTER PPMQULONG;</code>
PPMQVOID	<code>typedef PMQVOID MQPOINTER PPMQVOID;</code>
Où <code>defined(MQ_64_BIT)</code> correspond à une plateforme 64 bits.	

Pour une description de la variable de macro MQPOINTER, voir «types de données», à la page 269 .

*Déclarations de type de données COBOL*

Tableau 459. Noms et représentations de type de données COBOL

Type de données	Représentation
MQBOOL	PIC S9(9) BINARY
MQBYTE	PIC X

Tableau 459. Noms et représentations de type de données COBOL (suite)

Type de données	Représentation
MQBYTE8	PIC X(8)
MQBYTE16	PIC X(16)
MQBYTE24	PIC X(24)
MQBYTE32	PIC X(32)
MQBYTE40	PIC X(40)
MQCHAR	PIC X
MQCHAR4	PIC X(4)
MQCHAR8	PIC X(8)
MQCHAR12	PIC X(12)
MQCHAR20	PIC X(20)
MQCHAR28	PIC X(28)
MQCHAR32	PIC X(32)
MQCHAR48	PIC X(48)
MQCHAR64	PIC X(64)
MQCHAR128	PIC X(128)
MQCHAR256	PIC X(256)

<i>Tableau 459. Noms et représentations de type de données COBOL (suite)</i>	
<b>Type de données</b>	<b>Représentation</b>
MQFLOAT32	USAGE COMP-1
MQFLOAT64	USAGE COMP-2
MQHCONN	Sous z/OS PIC S9(9) COMP-5 Sur les autres plateformes PIC S9(9) BINARY
MQHOBJ	PIC S9(9) BINARY
MQINT8	PIC S9(2) BINARY
MQINT16	PIC S9(4) BINARY
MQINT64	PIC S9(18) BINARY
MQLONG	PIC S9(9) BINARY
MQPTR	POINTER
MQUINT8	PIC 9(2) BINARY
MQUINT16	PIC 9(4) BINARY
MQUINT64	PIC 9(18) BINARY
MQULONG	PIC 9(9) BINARY

*Déclarations de type de données PL/I*

Tableau 460. Noms et représentations des types de données PL/I

Type de données	Représentation
MQBOOL	fixed bin(31)
MQBYTE	char(1)
MQBYTE8	char(8)
MQBYTE16	char(16)
MQBYTE24	char(24)
MQBYTE32	char(32)
MQBYTE40	char(40)
MQCHAR	char(1)
MQCHAR4	char(4)
MQCHAR8	char(8)
MQCHAR12	char(12)
MQCHAR20	char(20)
MQCHAR28	char(28)
MQCHAR32	char(32)
MQCHAR48	char(48)
MQCHAR64	char(64)

Tableau 460. Noms et représentations des types de données PL/I (suite)

Type de données	Représentation
MQCHAR128	char(128)
MQCHAR256	char(256)
MQFLOAT32	binary float(21) ieee
MQFLOAT64	binary float(52) ieee
MQHCONN	fixed bin(31)
MQHOBJ	fixed bin(31)
MQINT8	fixed bin(7)
MQINT16	fixed bin(15)
MQINT64	fixed bin(63)
MQLONG	fixed bin(31)
MQPTR	pointer
MQUINT8	fixed bin(8)
MQUINT16	fixed bin(16)
MQUINT64	fixed bin(64)
MQULONG	fixed bin(32)

*Déclarations de type de données High Level Assembler*

Tableau 461. Noms et représentations des types de données assembleur System/390

Type de données	Représentation
MQBOOL	DS F
MQBYTE	DS XL1
MQBYTE8	DS XL8
MQBYTE16	DS XL16
MQBYTE24	DS XL24
MQBYTE32	DS XL32
MQBYTE40	DS XL40
MQCHAR	DS CL1
MQCHAR4	DS CL4
MQCHAR8	DS CL8
MQCHAR12	DS CL12
MQCHAR20	DS CL20
MQCHAR28	DS CL28
MQCHAR32	DS CL32
MQCHAR48	DS CL48
MQCHAR64	DS CL64

Tableau 461. Noms et représentations des types de données assembleur System/390 (suite)

Type de données	Représentation
MQCHAR128	DS CL128
MQCHAR256	DS CL256
MQFLOAT32	DS EB
MQFLOAT64	DS DB
MQHCONN	DS F
MQHOBJ	DS F
MQINT8	DS XL1
MQINT16	DS H
MQINT64	DS D
MQLONG	DS F
MQPTR	DS F
MQUINT8	DS XL1
MQUINT16	DS H
MQUINT64	DS D
MQULONG	DS F

### **Types de données de structure**

Récapitulatif des types de données de structure, des règles de mappage des structures MQI de manière cohérente et des conventions utilisées dans chaque description de type de données de structure.



- «Récapitulatif des types de données de structure utilisés sur les appels MQI ou les fonctions d'exit», à la page 265
- «Récapitulatif des types de données de structure utilisés dans les données de message», à la page 266
- «Règles de mappage des structures MQI de manière cohérente», à la page 266
- «Conventions utilisées dans chaque description de type de données de structure», à la page 267

## Récapitulatif des types de données de structure utilisés sur les appels MQI ou les fonctions d'exit

<i>Tableau 462. Types de données de structure utilisés sur les appels MQI ou les fonctions d'exit</i>		
<b>Structure</b>	<b>Description</b>	<b>Appels utilisés</b>
<a href="#">MQACH</a>	En-tête de chaîne d'exit d'API	
<a href="#">MQAIR</a>	Enregistrement d'informations d'authentification	<a href="#">MQCONN</a>
<a href="#">MQAXC</a>	Contexte d'exit d'API	
<a href="#">MQAXP</a>	Paramètre d'exit API	
<a href="#">MQBMHO</a>	Options de traitement de la mémoire tampon vers le message	<a href="#">MQBUFMH</a>
<a href="#">MQBO</a>	Options de début	<a href="#">MQBEGIN</a>
<a href="#">MQCBD</a>	Descripteur de rappel	<a href="#">MQCB</a>
<a href="#">MQCBO</a>	Options de création de sac	Sac mqCreate
<a href="#">MQCHARV</a>	Chaîne de longueur variable	<a href="#">MQINQMP</a>
<a href="#">MQCNO</a>	Options de connexion	<a href="#">MQCONN</a>
<a href="#">MQCSP</a>	Paramètres de sécurité	<a href="#">MQCONN</a>
<a href="#">MQCTL</a>	Options de rappel	<a href="#">MQCTL</a>
<a href="#">MQDMPO</a>	Options de suppression de propriété de message	<a href="#">MQDLTMP</a>
<a href="#">MQGMO</a>	Options de message Get	<a href="#">MQGet</a>
<a href="#">MQIMPO</a>	Options de propriété de message d'interrogation	<a href="#">MQINQMP</a>
<a href="#">MQMD</a>	Descripteur de message	<a href="#">MQBUFMH</a> , <a href="#">MQMHBUF</a> , <a href="#">MQCB</a> , <a href="#">MQGET</a> , <a href="#">MQPUT</a> , <a href="#">MQPUT1</a>
<a href="#">MQMHBO</a>	Options de traitement des messages vers la mémoire tampon	<a href="#">MQMHBUF</a>
<a href="#">MQOD</a>	Descripteur d'objet	<a href="#">MQOPEN</a> , <a href="#">MQPUT1</a>
<a href="#">MQOR</a>	Enregistrement d'objet	<a href="#">MQOPEN</a> , <a href="#">MQPUT1</a>
<a href="#">MQPD</a>	Descripteur de propriété	<a href="#">MQSETMP</a>
<a href="#">MQPMO</a>	Options de message Put	<a href="#">MQPUT</a> , <a href="#">MQPUT1</a>
<a href="#">MQPMR</a>	Enregistrement d'insertion de message	<a href="#">MQPUT</a> , <a href="#">MQPUT1</a>
<a href="#">MQRR</a>	Enregistrement de réponse	<a href="#">MQOPEN</a> , <a href="#">MQPUT</a> , <a href="#">MQPUT1</a>

Structure	Description	Appels utilisés
<a href="#">MQSCO</a>	Options de configuration TLS	<a href="#">MQCONN</a>
<a href="#">MQSD</a>	Descripteur d'abonnement	<a href="#">MQSUB</a>
<a href="#">MQSMPO</a>	Définir l'option de propriété de message	<a href="#">MQSETMP</a>
<a href="#">MQSRO</a>	Options de demande d'abonnement	<a href="#">MQSUBRQ</a>
<a href="#">MQSTS</a>	Structure de génération de rapports de statut	<a href="#">MQSTAT</a>

## Récapitulatif des types de données de structure utilisés dans les données de message

Structure	Description
<a href="#">MQCIH</a>	En-tête d'informations CICS
<a href="#">MQCFH</a>	En-tête PCF
<a href="#">MQEPH</a>	En-tête PCF imbriqué
<a href="#">MQDH</a>	En-tête de distribution
<a href="#">MQDLH</a>	En-tête de rebut (message non distribué)
<a href="#">MQIIH</a>	En-tête d'informations IMS
<a href="#">MQMDE</a>	Extension du descripteur de message
<a href="#">MQRFH</a>	Règles et en-tête de formatage
<a href="#">MQRFH2</a>	Règles et en-tête de formatage 2
<a href="#">MQRMH</a>	En-tête de message de référence
<a href="#">MQTM</a>	Message de déclenchement
<a href="#">MQTMC2</a>	Message de déclenchement (format de caractère 2)
<a href="#">MQWIH</a>	En-tête d'informations de travail
<a href="#">MQXQH</a>	En-tête de file d'attente de transmission

**Remarque :** La structure MQDXP (paramètre d'exit de conversion de données) est décrite dans «Exit de conversion de données», à la page 951, avec les appels de conversion de données associés.

## Règles de mappage des structures MQI de manière cohérente

Les langages de programmation varient dans leur niveau de prise en charge des structures, et certaines règles et conventions sont adoptées pour mapper les structures MQI de manière cohérente dans chaque langage de programmation:

1. Les structures doivent être alignées sur leurs limites naturelles.
  - La plupart des structures MQI nécessitent un alignement sur 4 octets.
  - Sous IBM i, les structures contenant des pointeurs requièrent un alignement sur 16 octets ; il s'agit de MQCNO, MQOD, MQPMO.

2. Chaque zone d'une structure doit être alignée sur sa limite naturelle.
  - Les zones dont les types de données sont équivalents à MQLONG doivent être alignées sur des limites de 4 octets.
  - Les zones dont les types de données sont équivalents à MQPTR doivent être alignées sur des limites de 16 octets sur IBM i et sur des limites de 4 octets dans d'autres environnements.
  - Les autres zones sont alignées sur des limites de 1 octet.
3. La longueur d'une structure doit être un multiple de son alignement de limite.
  - La plupart des structures MQI ont des longueurs multiples de 4 octets.
  - Sous IBM i, les structures contenant des pointeurs ont des longueurs multiples de 16 octets.
4. Si nécessaire, des octets ou des zones de remplissage doivent être ajoutés pour garantir la conformité avec les règles précédentes.

## Conventions utilisées dans chaque description de type de données de structure

La description de chaque type de données de structure comprend:

- Un aperçu de l'objectif et de l'utilisation de la structure
- Description des zones de la structure, sous une forme indépendante du langage de programmation
- Exemples de déclaration de la structure dans chacun des langages de programmation pris en charge

La description de chaque type de données de structure contient les sections suivantes:

### Nom de structure

Nom de la structure, suivi d'un récapitulatif des zones de la structure.

### Présentation

Brève description de l'objet et de l'utilisation de la structure.

### Zones

Description des zones. Pour chaque champ, le nom du champ est suivi de son type de données élémentaires entre parenthèses (). Dans le texte, les noms de zone sont affichés en italique ; par exemple, *Version*.

Il contient également une description de l'objectif de la zone, ainsi qu'une liste de toutes les valeurs que la zone peut prendre. Les noms des constantes sont affichés en majuscules ; par exemple, MQGMO\_STRUC\_ID. Un ensemble de constantes ayant le même préfixe est affiché à l'aide du caractère \*, par exemple: MQIA\_\*

Dans les descriptions des zones, les termes suivants sont utilisés:

#### entrée

Vous fournissez des informations dans la zone lorsque vous effectuez un appel.

#### sortie

Le gestionnaire de files d'attente renvoie des informations dans la zone lorsque l'appel se termine ou échoue.

#### d'entrée-sortie

Vous fournissez des informations dans la zone lorsque vous effectuez un appel et le gestionnaire de files d'attente modifie les informations lorsque l'appel se termine ou échoue.

### Valeurs initiales

Tableau présentant les valeurs initiales de chaque zone dans les fichiers de définition de données fournis avec l'interface MQI.

### Déclaration C

Déclaration typique de la structure en C.

### Déclaration COBOL

Déclaration standard de la structure en COBOL.

### Déclaration PL/I

Déclaration typique de la structure en PL/I.

## Déclaration High Level Assembler

Déclaration standard de la structure dans le langage assembleur System/390 .

## Déclaration Visual Basic



Déclaration typique de la structure dans Visual Basic.

## Programmation C

Informations pour vous aider à utiliser l'interface MQI du langage de programmation C.

- «Fichiers d'en-tête», à la page 268
- «Fonctions», à la page 268
- «Paramètres avec type de données non défini», à la page 269
- «types de données», à la page 269
- «Manipulation des chaînes binaires», à la page 269
- «Manipulation des chaînes de caractères», à la page 269
- «Valeurs initiales pour les structures», à la page 270
- «Valeurs initiales pour les structures dynamiques», à la page 270
- «Utiliser à partir de C++», à la page 271
- «Conventions de notation», à la page 271

## Fichiers d'en-tête

Tableau 464. Fichiers d'en-tête C	
Fichier	Contenu
CMQC	Prototypes de fonction, types de données et constantes nommées pour l'interface MQI principale
CMQXC	Prototypes de fonction, types de données et constantes nommées pour l'exit de conversion de données
CMQEC	Prototypes de fonction, types de données et constantes nommées pour l'interface MQI principale, l'exit de conversion de données et la structure des points d'entrée d'interface (CMQEC inclut CMQXC et CMQC).
CMQSTRC	Fonctions qui convertissent les définitions de constante MQI en équivalents texte.  <b>Avertissement :</b>  Applicable à z/OS à partir de IBM MQ 9.1. Les programmes utilisant ce fichier d'en-tête doivent être compilés avec l'option de compilation LONGNAME.

Pour améliorer la portabilité des applications, codez le nom du fichier d'en-tête en minuscules sur la directive de préprocesseur `#include` :

```
#include "cmqec.h"
```

## Fonctions

Il n'est pas nécessaire de spécifier tous les paramètres transmis par l'adresse chaque fois que vous appelez une fonction.

- Transmettez les paramètres *d'entrée uniquement* et de type MQHCONN, MQHOBJ ou MQLONG par valeur.
- Transmettez tous les autres paramètres par adresse.

Lorsqu'un paramètre particulier n'est pas requis, utilisez un pointeur null comme paramètre sur l'appel de fonction, à la place de l'adresse des données de paramètre. Les paramètres pour lesquels cela est possible sont identifiés dans les descriptions d'appel.

Aucun paramètre n'est renvoyé comme valeur de la fonction ; dans la terminologie C, cela signifie que toutes les fonctions renvoient `void`.

Les attributs de la fonction sont définis par la variable de macro `MQENTRY` ; la valeur de cette variable de macro dépend de l'environnement.

## Paramètres avec type de données non défini

Le paramètre **Buffer** des fonctions `MQGET`, `MQPUT` et `MQPUT1` a un type de données non défini. Ce paramètre est utilisé pour envoyer et recevoir les données de message de l'application.

Les paramètres de ce type sont présentés dans les exemples C sous la forme de tableaux de `MQBYTE`. Vous pouvez déclarer les paramètres de cette manière, mais il est généralement plus pratique de les déclarer comme la structure particulière qui décrit la présentation des données dans le message. Déclarez le paramètre de fonction réel en tant que pointeur vers vide et indiquez l'adresse de n'importe quel type de données en tant que paramètre lors de l'appel de fonction.

## types de données

Définissez tous les types de données à l'aide de l'instruction C `typedef`. Pour chaque type de données, définissez également le type de données de pointeur correspondant. Le nom du type de données de pointeur est le nom du type de données élémentaire ou de structure précédé de la lettre P pour désigner un pointeur. Définissez les attributs du pointeur à l'aide de la variable de macro `MQPOINTER` ; la valeur de cette variable de macro dépend de l'environnement. L'exemple suivant montre comment déclarer des types de données de pointeur:

```
#define MQPOINTER *           /* depends on environment */
...
typedef MQLONG MQPOINTER PMQLONG; /* pointer to MQLONG */
typedef MQMD MQPOINTER PMQMD; /* pointer to MQMD */
```

## Manipulation des chaînes binaires

Déclarez des chaînes de données binaires comme l'un des types de données `MQBYTEn`.

Chaque fois que vous copiez, comparez ou définissez des zones de ce type, utilisez les fonctions C **`memcpy`**, **`memcmp`** ou **`memset`** ; par exemple:

```
#include <string.h>
#include "cmqc.h"

MQMD MyMsgDesc;

memcpy(MyMsgDesc.MsgId,          /* set "MsgId" field to nulls */
       MQMI_NONE,               /* ...using named constant */
       sizeof(MyMsgDesc.MsgId));

memset(MyMsgDesc.CorrelId,      /* set "CorrelId" field to nulls */
       0x00,                   /* ...using a different method */
       sizeof(MQBYTE24));
```

N'utilisez pas les fonctions de chaîne **`strcpy`**, **`strcmp`**, **`strncpy`** ou **`strncmp`**, car elles ne fonctionnent pas correctement pour les données déclarées avec les types de données `MQBYTEn`.

## Manipulation des chaînes de caractères

Lorsque le gestionnaire de files d'attente renvoie des données de type caractère à l'application, il remplit toujours les données de type caractère avec des blancs sur la longueur définie de la zone. Le gestionnaire de files d'attente *ne renvoie pas* de chaînes à terminaison nulle.

Par conséquent, lors de la copie, de la comparaison ou de la concaténation de ces chaînes, utilisez les fonctions de chaîne **strncpy**, **strncmp** ou **strncat**.

N'utilisez pas les fonctions de chaîne qui nécessitent que la chaîne se termine par une valeur nulle (**strcpy**, **strcmp**, **strcat**). En outre, n'utilisez pas la fonction **strlen** pour déterminer la longueur de la chaîne ; utilisez plutôt la fonction **sizeof** pour déterminer la longueur de la zone.

## Valeurs initiales pour les structures

Les fichiers d'en-tête définissent diverses variables de macro que vous pouvez utiliser pour fournir des valeurs initiales pour les structures MQ lorsque vous déclarez des instances de ces structures.

Ces variables macro ont des noms de la forme MQxxx\_DEFAULT, où MQxxx représente le nom de la structure. Ils sont utilisés de la manière suivante:

```
MQMD   MyMsgDesc = {MQMD_DEFAULT};
MQPMO  MyPutOpts = {MQPMO_DEFAULT};
```

Pour certaines zones de caractères (par exemple, les zones *StrucId* qui apparaissent dans la plupart des structures ou la zone *Format* qui apparaît dans MQMD), l'interface MQI définit des valeurs particulières qui sont valides. Pour chacune des valeurs valides, deux variables macro sont fournies:

- Une variable macro définit la valeur sous la forme d'une chaîne avec une longueur, en excluant les correspondances nulles implicites, exactement la longueur définie de la zone. Par exemple, pour la zone *Format* dans MQMD, la variable de macro suivante est fournie (↵ représente un caractère blanc unique):

```
#define MQFMT_STRING "MQSTR↵↵↵"
```

Utilisez ce formulaire avec les fonctions `memcpy` et `memcmp`.

- L'autre variable de macro définit la valeur comme un tableau de caractères ; le nom de cette variable de macro est le nom de la forme de chaîne suffixé avec `_ARRAY`. Exemple :

```
#define MQFMT_STRING_ARRAY 'M','Q','S','T','R','↵','↵','↵'
```

Utilisez ce formulaire pour initialiser la zone lorsque vous déclarez une instance de la structure avec des valeurs différentes de celles fournies par la variable macro MQMD\_DEFAULT. (Ce n'est pas toujours nécessaire ; dans certains environnements, vous pouvez utiliser la forme de chaîne de la valeur dans les deux situations. Toutefois, vous pouvez utiliser le formulaire de tableau pour les déclarations, car cela est requis pour la compatibilité avec le langage de programmation C + +.)

## Valeurs initiales pour les structures dynamiques

Lorsqu'un nombre variable d'instances d'une structure est requis, les instances sont généralement créées dans la mémoire principale obtenue dynamiquement à l'aide des fonctions `calloc` ou `malloc`. Pour initialiser les zones dans de telles structures, tenez compte de la technique suivante:

1. Déclarez une instance de la structure à l'aide de la variable de macro MQxxx\_DEFAULT appropriée pour initialiser la structure. Cette instance devient le modèle pour les autres instances:

```
MQMD Model = {MQMD_DEFAULT}; /* declare model instance */
```

Les mots clés `static` ou `auto` peuvent être codés sur la déclaration afin de donner à l'instance de modèle une durée de vie statique ou dynamique, selon les besoins.

- Utilisez les fonctions `calloc` ou `malloc` pour obtenir du stockage pour une instance dynamique de la structure:

```
PMQMD Instance;  
Instance = malloc(sizeof(MQMD)); /* get storage for dynamic instance */
```

- Utilisez la fonction `memcpy` pour copier l'instance de modèle dans l'instance dynamique:

```
memcpy(Instance,&Model,sizeof(MQMD)); /* initialize dynamic instance */
```

## Utiliser à partir de C++

Pour le langage de programmation C + +, les fichiers d'en-tête contiennent les instructions supplémentaires suivantes qui sont incluses uniquement lorsque vous utilisez un compilateur C + +:

```
#ifdef __cplusplus  
extern "C" {  
#endif  
  
/* rest of header file */  
  
#ifdef __cplusplus  
}  
#endif
```

## Conventions de notation

Ces informations montrent comment appeler les fonctions et déclarer des paramètres.

Dans certains cas, les paramètres sont des tableaux dont la taille n'est pas fixe. Pour ceux-ci, un `n` minuscule est utilisé pour représenter une constante numérique. Lorsque vous codez la déclaration pour ce paramètre, remplacez le `n` par la valeur numérique requise.

## Programmation COBOL

Informations pour vous aider à utiliser l'interface MQI du langage de programmation COBOL.

- [«Copier des fichiers»](#), à la page 271
- [«Structures»](#), à la page 272
- [«Pointeurs»](#), à la page 273
- [«Constante nommée»](#), à la page 273
- [«Conventions de notation»](#), à la page 274

## Copier des fichiers

Divers fichiers COPY sont fournis pour vous aider à écrire des programmes d'application COBOL qui utilisent l'interface MQI. Il existe deux fichiers contenant des constantes nommées et deux fichiers pour chacune des structures.

Chaque structure est fournie sous deux formes: une forme avec des valeurs initiales, et une forme sans:

- Utilisez les structures avec les valeurs initiales de la SECTION WORKING-STORAGE d'un programme COBOL ; elles sont contenues dans des fichiers COPY dont les noms sont suffixés par la lettre V (pour les valeurs).
- Utilisez les structures sans valeurs initiales dans la LINKAGE SECTION d'un programme COBOL ; elles sont contenues dans des fichiers COPY dont les noms sont suffixés par la lettre L (pour Linkage).

Les fichiers COPY sont récapitulés dans le tableau suivant. Tous les fichiers répertoriés ne sont pas disponibles dans tous les environnements.

Tableau 465. Fichiers COBOL COPY

Fichier (avec les valeurs initiales)	Fichier (sans valeurs initiales)	Contenu
CMQAIRV	CMQAIRL	Enregistrement d'informations d'authentification
CMQBOV	CMQBOL	Structure des options de début
CMQCIHV	CMQCIHL	Structure d'en-tête d'informations CICS
CMQCNNOV	CMQCNOL	Structure des options de connexion
CMQDHOV	CMQDHL	Structure d'en-tête de distribution
CMQDLHOV	CMQDLHL	Structure d'en-tête de rebut
CMQDXPOV	CMQDXPL	Structure des paramètres d'exit de conversion de données
CMQGMHOV	CMQGMOL	Obtenir la structure des options de message
CMQIIHOV	CMQIIHL	Structure d'en-tête d'informations IMS
CMQMDHOV	CMQMDL	Structure de descripteur de message
CMQMDEHOV	CMQMDEL	Structure d'extension du descripteur de message
CMQMD1HOV	CMQMD1L	Structure de descripteur de message version 1
CMQODHOV	CMQODL	Structure de descripteur d'objet
CMQORHOV	CMQORL	Structure d'enregistrement d'objet
CMQPMHOV	CMQPMOL	Structure des options de message d'insertion
CMQRFHOV	CMQRFHL	Règles et structure d'en-tête de formatage
CMQRFH2HOV	CMQRFH2L	Règles et formatage de la structure d'en-tête version 2
CMQRMHOV	CMQRMHL	Structure d'en-tête de message de référence
CMQRRHOV	CMQRRL	Structure d'enregistrement de réponse
CMQSCHOV	CMQSCOL	Options de configuration TLS
CMQTMHOV	CMQTML	Structure des messages de déclenchement
CMQTMCHOV	CMQTMCL	Structure du message de déclenchement (format de caractères)
CMQTMCH2HOV	CMQTMCH2L	Structure du message de déclenchement (format de caractère) version 2
CMQWIHOV	CMQWIHL	Structure d'en-tête des informations de travail
CMQXQHOV	CMQXQHL	Structure d'en-tête de file d'attente de transmission
CMQV	-	Constantes nommées pour l'interface MQI principale
CMQXV	-	Constantes nommées pour l'exit de conversion de données
CMQMD2HOV	CMQMD2L	Structure de descripteur de message version 2

## Structures

Dans le fichier COPY, chaque déclaration de structure commence par un élément level-10 ; cela vous permet de déclarer plusieurs instances de la structure en codant la déclaration level-01 , puis en utilisant



l'instruction COPY pour copier le reste de la déclaration de structure. Pour faire référence à l'instance appropriée, utilisez le mot clé IN :

```
* Declare two instances of MQMD
01 MY-MQMD.
   COPY CMQMDV.
01 MY-OTHER-MQMD.
   COPY CMQMDV.
*
* Set MSGTYPE field in MY-OTHER-MQMD
  MOVE MQMT-REQUEST TO MQMD-MSGTYPE IN MY-OTHER-MQMD.
```

Alignez les structures sur les limites appropriées. Si vous utilisez l'instruction COPY pour inclure une structure à la suite d'un élément qui n'est pas l'élément level-01, assurez-vous que la structure commence au décalage approprié à partir du début de l'élément level-01. La plupart des structures MQI nécessitent un alignement à 4 octets ; les exceptions à cela sont MQCNO, MQOD et MQPMO, qui requièrent un alignement à 16 octets sur IBM i.

Dans cette section, les noms des zones des structures sont affichés sans préfixe. En COBOL, les noms de zone sont précédés du nom de la structure suivi d'un trait d'union. Toutefois, si le nom de la structure se termine par un chiffre, indiquant que la structure est une deuxième version ou une version ultérieure de la structure d'origine, le chiffre numérique est omis du préfixe. Les noms de zone en COBOL sont affichés en majuscules (bien que des minuscules ou une casse mixte puissent être utilisées si nécessaire). Par exemple, la zone *MsgType* décrite dans «MQMD-Descripteur de message», à la page 438 devient MQMD-MSGTYPE en COBOL.

Les structures de suffixe V sont déclarées avec des valeurs initiales pour toutes les zones ; vous devez définir uniquement les zones dans lesquelles vous souhaitez une valeur différente de la valeur initiale fournie.

## Pointeurs

Certaines structures doivent traiter des données facultatives qui peuvent être discontiguës à la structure, telles que les enregistrements MQOR et MQRR traités par la structure MQOD.

Pour traiter ces données facultatives, les structures contiennent des zones qui sont déclarées avec le type de données de pointeur. Cependant, COBOL ne prend pas en charge le type de données de pointeur dans tous les environnements. Pour cette raison, les données facultatives peuvent également être traitées à l'aide de zones qui contiennent le décalage des données à partir du début de la structure.

Si vous souhaitez porter une application entre des environnements, déterminez si le type de données de pointeur est disponible dans tous les environnements prévus. Si ce n'est pas le cas, l'application doit traiter les données facultatives à l'aide des zones de décalage au lieu des zones de pointeur.

Dans les environnements où les pointeurs ne sont pas pris en charge, déclarez les zones de pointeur en tant que chaînes d'octets de la longueur appropriée, la valeur initiale étant la chaîne d'octets tout-null. Ne modifiez pas cette valeur initiale si vous utilisez les zones de décalage.

## Constante nommée

Dans ces informations, les noms des constantes sont affichés avec le caractère de soulignement (  ) dans le nom. En COBOL, utilisez le trait d'union (-) à la place du trait de soulignement.

Les constantes qui ont des valeurs de chaîne de caractères utilisent le guillemet simple comme délimiteur de chaîne (''). Dans certains environnements, vous devrez peut-être spécifier une option de compilateur appropriée pour que le compilateur accepte le guillemet simple comme délimiteur de chaîne à la place du guillemet double.

Les constantes nommées sont déclarées dans les fichiers COPY en tant qu'éléments level-10. Pour utiliser les constantes, déclarez explicitement l'élément level-01, puis utilisez l'instruction COPY pour copier les déclarations des constantes:

```
* Declare a structure to hold the constants
```

```
01 MY-MQ-CONSTANTS.  
COPY CMQV.
```

La méthode précédente permet aux constantes d'occuper la mémoire du programme même si elles ne sont pas référencées. Si vous incluez les constantes dans de nombreux programmes distincts au sein d'une même unité d'exécution, il existe plusieurs copies des constantes, ce qui consomme inutilement de la mémoire principale. Evitez cet effet en utilisant l'une des techniques suivantes:

- Ajoutez la clause GLOBAL à la déclaration level-01 :

```
* Declare a global structure to hold the constants  
01 MY-MQ-CONSTANTS GLOBAL.  
COPY CMQV.
```

Permet d'allouer de la mémoire pour un seul ensemble de constantes au sein de l'unité d'exécution. Toutefois, les constantes peuvent être référencées par n'importe quel programme au sein de l'unité d'exécution, et pas seulement par le programme qui contient la déclaration level-01 .

**Remarque :** La clause GLOBAL n'est pas prise en charge dans tous les environnements.

- Copiez manuellement dans chaque programme uniquement les constantes référencées par ce programme. N'utilisez pas l'instruction COPY pour copier toutes les constantes dans le programme.

## Conventions de notation

Les sections précédentes de cette rubrique montrent comment appeler des appels et déclarer des paramètres. Dans certains cas, les paramètres sont des tables ou des chaînes de caractères dont la taille n'est pas fixe. Pour ceux-ci, un n minuscule est utilisé pour représenter une constante numérique. Lorsque vous codez la déclaration pour ce paramètre, remplacez le n par la valeur numérique requise.

### Programmation High Level Assembler

Informations pour vous aider à utiliser l'interface MQI du langage de programmation System/390 Assembler.

- «Macros», à la page 274
- «Structures», à la page 275
- «Macro MQVERA», à la page 275
- «Conventions de notation», à la page 275

## Macros

Il existe deux macros pour les constantes nommées et une macro pour chacune des structures. Ces fichiers sont résumés dans le tableau suivant.

Fichier	Contenu
CMQA	Constantes nommées (égales) pour l'interface MQI principale
CMQCIHA	Structure d'en-tête d'informations CICS
CMQCNOA	Structure des options de connexion
CMQDLHA	Structure d'en-tête de rebut
CMQDXPA	Structure des paramètres d'exit de conversion de données
CMQGMOA	Obtenir la structure des options de message
CMQIIHA	Structure d'en-tête d'informations IMS
CMQMDA	Structure de descripteur de message

Tableau 466. Macros assembleur (suite)

Fichier	Contenu
CMQMDEA	Structure d'extension du descripteur de message
CMQODA	Structure de descripteur d'objet
CMQPMOA	Structure des options de message d'insertion
CMQRFHA	Règles et structure d'en-tête de formatage
CMQRFH2A	Règles et formatage de la structure d'en-tête version 2
CMQRMHA	Structure d'en-tête de message de référence
Intervalle CMQTMA	Structure des messages de déclenchement
CMQTMCA	Structure du message de déclenchement (format de caractère) version 2
CMQVERA	Contrôle de version de structure
CMQWIHA	Structure d'en-tête des informations de travail
CMQXA	Constantes nommées pour l'exit de conversion de données
CMQXPA	Structure des paramètres d'exit de croisement d'API
CMQXQHA	Structure d'en-tête de file d'attente de transmission

## Structures

Les structures sont générées par des macros qui ont différents paramètres pour contrôler l'action de la macro. Voir «Structures», à la page 275

### Macro MQVERA

Cette macro permet de définir la valeur par défaut à utiliser pour le paramètre DCLVER dans les macros de structure.

La valeur spécifiée par CMQVERA est utilisée par la macro de structure uniquement si vous omettez le paramètre DCLVER dans l'appel de la macro de structure. La valeur par défaut est définie en codant la macro CMQVERA avec le paramètre DCLVER :

#### **DCLVER=EN COURS**

La version par défaut est définie sur la version en cours (la plus récente).

#### **DCLVER=SPECIFIE**

La version par défaut est définie sur la version spécifiée par le paramètre VERSION .

Vous devez spécifier le paramètre **DCLVER** et la valeur doit être en majuscules. La valeur définie par CMQVERA reste la valeur par défaut jusqu'au prochain appel de CMQVERA ou jusqu'à la fin de l'assemblage. Si vous omettez CMQVERA, la valeur par défaut est DCLVER=CURRENT.

## Conventions de notation

D'autres rubriques expliquent comment appeler les appels et déclarer des paramètres. Dans certains cas, les paramètres sont des tableaux ou des chaînes de caractères dont la taille n'est pas fixe et pour lesquels un n minuscule est utilisé pour représenter une constante numérique. Lorsque vous codez la déclaration pour ce paramètre, remplacez le n par la valeur numérique requise.

### Structures

Les structures sont générées par des macros qui ont différents paramètres pour contrôler l'action de la macro.

**Remarque :** De temps à autre, de nouvelles versions des structures IBM MQ sont introduites. Les zones supplémentaires d'une nouvelle version peuvent faire en sorte qu'une structure de moins de 256 octets devienne supérieure à 256 octets. Pour cette raison, écrivez des instructions d'assembleur destinées à copier une structure IBM MQ ou à définir une structure IBM MQ sur des valeurs nulles, afin de fonctionner correctement avec des structures pouvant dépasser 256 octets. Vous pouvez également utiliser le paramètre de macro DCLVER ou la macro CMQVERA avec le paramètre VERSION pour déclarer une version spécifique de la structure.

- «Spécification du nom de la structure», à la page 276
- «Spécification de la forme de la structure», à la page 276
- «Contrôle de la version de la structure», à la page 276
- «Déclaration d'une structure imbriquée dans une autre», à la page 277
- «Spécification de valeurs initiales pour les champs», à la page 277
- «Contrôle de la liste», à la page 277

## Spécification du nom de la structure

Pour déclarer plusieurs instances d'une structure, la macro préfixe le nom de chaque zone de la structure avec une chaîne spécifique à l'utilisateur et un trait de soulignement.

La chaîne utilisée est le libellé spécifié lors de l'appel de la macro. Si aucun libellé n'est spécifié, le nom de la structure est utilisé pour construire le préfixe:

```
* Declare two object descriptors
      CMQODA ,          Prefix used="MQOD_" (the default)
MY_MQOD CMQODA ,      Prefix used="MY_MQOD_"
```

Les déclarations de structure affichées dans cette section utilisent le préfixe par défaut.

## Spécification de la forme de la structure

Les déclarations de structure peuvent être générées par la macro sous l'une des deux formes suivantes, contrôlées par le paramètre DSECT :

### DSECT = OUI

Une instruction DSECT assembleur est utilisée pour démarrer une nouvelle section de données ; la définition de structure suit immédiatement l'instruction DSECT . Le libellé de l'appel de macro est utilisé comme nom de la section de données ; si aucun libellé n'est spécifié, le nom de la structure est utilisé.

### DSECT = NON

Les instructions DC de l'assembleur permettent de définir la structure à la position en cours dans la routine. Les zones sont initialisées avec des valeurs, qui peuvent être spécifiées en codant les paramètres appropriés lors de l'appel de la macro. Les zones pour lesquelles aucune valeur n'est spécifiée dans l'appel de macro sont initialisées avec des valeurs par défaut.

La valeur indiquée doit être en majuscules. Si le paramètre DSECT n'est pas spécifié, DSECT = NO est utilisé par défaut.

## Contrôle de la version de la structure

Par défaut, les macros déclarent toujours la version la plus récente de chaque structure.

Bien que vous puissiez utiliser le paramètre de macro VERSION pour spécifier une valeur pour la zone *Version* dans la structure, ce paramètre définit la valeur initiale de la zone *Version* et ne contrôle pas la version de la structure réellement déclarée. Pour contrôler la version de la structure déclarée, utilisez le paramètre DCLVER :

### DCLVER=EN COURS

La version déclarée est la version en cours (la plus récente).

## DCLVER=SPECIFIE

La version déclarée est la version spécifiée par le paramètre VERSION . Si vous omettez le paramètre VERSION , la version par défaut est la version 1.

Si vous spécifiez le paramètre VERSION , la valeur doit être une constante numérique auto-définie ou la constante nommée pour la version requise (par exemple, MQCNO\_VERSION\_3). Si vous spécifiez une autre valeur, la structure est déclarée comme si DCLVER=CURRENT avait été spécifié, même si la valeur de VERSION est résolue en une valeur valide.

La valeur indiquée doit être en majuscules. Si vous omettez le paramètre DCLVER , la valeur utilisée est extraite de la variable macro globale MQDCLVER . Vous pouvez définir cette variable à l'aide de la macro CMQVERA.

## Déclaration d'une structure imbriquée dans une autre

Pour déclarer une structure en tant que composant d'une autre structure, utilisez le paramètre IMBRIQUÉ :

### NESTED=YES

La déclaration de structure est imbriquée dans une autre.

### NESTED=NO

La déclaration de structure n'est pas imbriquée dans une autre.

La valeur indiquée doit être en majuscules. Si vous omettez le paramètre IMBRIQUÉ , NESTED=NO est utilisé.

## Spécification de valeurs initiales pour les champs

Indiquez la valeur à utiliser pour initialiser une zone dans une structure en codant le nom de cette zone (sans le préfixe) en tant que paramètre dans l'appel de macro, accompagné de la valeur requise.

Par exemple, pour déclarer une structure de descripteur de message avec la zone *MsgType* initialisée avec MQMT\_REQUEST et la zone *ReplyToQ* initialisée avec la chaîne "MY\_REPLY\_TO\_QUEUE", utilisez ce qui suit:

```
MY_MQMD  CMQMDA  MSGTYPE=MQMT_REQUEST,          X
          REPLYTOQ=MY_REPLY_TO_QUEUE
```

Si vous spécifiez une constante nommée (égale) comme valeur lors de l'appel de la macro, utilisez la macro CMQA pour définir la constante nommée. Ne placez pas les valeurs de chaîne de caractères entre apostrophes.

## Contrôle de la liste

Contrôlez l'apparence de la déclaration de structure dans la liste de l'assembleur à l'aide du paramètre LIST :

### LISTE = OUI

La déclaration de structure apparaît dans la liste de l'assembleur.

### LISTE = NON

La déclaration de structure n'apparaît pas dans la liste de l'assembleur.

La valeur indiquée doit être en majuscules. Si vous omettez le paramètre LIST , LIST = NO est utilisé.

## MQAIR-Enregistrement des informations d'authentification

La structure MQAIR permet à une application s'exécutant en tant qu' IBM MQ MQI client de spécifier des informations sur un authentificateur à utiliser pour la connexion client. La structure est un paramètre d'entrée dans l'appel MQCONN.

## Disponibilité

La structure MQAIR est disponible pour les clients suivants:

-  AIX
-  Linux
-  Windows

## Jeu de caractères et codage

Les données de MQAIR doivent être dans le jeu de caractères et le codage du gestionnaire de files d'attente local ; elles sont fournies par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et MQENC\_NATIVE.

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Tableau 467. Zones dans MQAIR		
Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID_STRUC_MQAIR_	'AIR↵'
<u>Version</u> (numéro de version de la structure)	MQAIR_VERSION_1	1
<u>AuthInfoType</u> (type des informations d'authentification)	MQAIT_CRL_LDAP	1
<u>AuthInfoConnName</u> (nom de connexion du serveur CRL LDAP)	Aucun	Chaîne nulle ou blancs
<u>LDAPUserNamePtr</u> (adresse du nom d'utilisateur LDAP)	Aucun	Pointeur null ou octets null
<u>LDAPUserNameDécalage</u> (décalage du nom d'utilisateur LDAP à partir du début de MQSCO)	Aucun	0
<u>LDAPUserNameLongueur</u> (longueur du nom d'utilisateur LDAP)	Aucun	0
<u>LDAPPassword</u> (mot de passe permettant d'accéder au serveur LDAP)	Aucun	Chaîne nulle ou blancs
<b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieure à MQAIR_VERSION_2.		
<u>OCSPResponderURL</u> (URL à laquelle le répondeur OCSP peut être contacté)	Aucun	Chaîne nulle ou blancs
<b>Remarques :</b> 1. Le symbole ↵ représente un caractère blanc unique. 2. Dans le langage de programmation C, la variable macroMQAIR_DEFAULT contient les valeurs répertoriées dans la table. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure: <pre>MQAIR MyAIR = {MQAIR_DEFAULT};</pre>		

## Déclarations de langue

### Déclaration C pour MQAIR

```
typedef struct tagMQAIR MQAIR;
struct tagMQAIR {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;          /* Structure version number */
    MQLONG     AuthInfoType;     /* Type of authentication
                                information */
    MQCHAR264  AuthInfoConnName; /* Connection name of CRL LDAP
                                server */
    PMQCHAR    LDAPUserNamePtr;  /* Address of LDAP user name */
    MQLONG     LDAPUserNameOffset; /* Offset of LDAP user name from start
                                of MQAIR structure */
    MQLONG     LDAPUserNameLength; /* Length of LDAP user name */
    MQCHAR32   LDAPPASSWORD;     /* Password to access LDAP server */
    MQCHAR256  OCSPResponderURL; /* URL of OCSP responder */
};
```

### Déclaration COBOL pour MQAIR

```
** MQAIR structure
10 MQAIR.
** Structure identifier
15 MQAIR-STRUCID PIC X(4).
** Structure version number
15 MQAIR-VERSION PIC S9(9) BINARY.
** Type of authentication information
15 MQAIR-AUTHINFOTYPE PIC S9(9) BINARY.
** Connection name of CRL LDAP server
15 MQAIR-AUTHINFOCONNNAME PIC X(264).
** Address of LDAP user name
15 MQAIR-LDAPUSERNAMEPTR POINTER.
** Offset of LDAP user name from start of MQAIR structure
15 MQAIR-LDAPUSERNAMEOFFSET PIC S9(9) BINARY.
** Length of LDAP user name
15 MQAIR-LDAPUSERNAMELENGTH PIC S9(9) BINARY.
** Password to access LDAP server
15 MQAIR-LDAPPASSWORD PIC X(32).
** URL of OCSP responder
15 MQAIR-OCSPRESPONDERURL PIC X(256).
```

### Déclaration Visual Basic pour MQAIR

```
Type MQAIR
    StrucId As String*4 'Structure identifier'
    Version As Long 'Structure version number'
    AuthInfoType As Long 'Type of authentication information'
    AuthInfoConnName As String*264 'Connection name of CRL LDAP server'
    LDAPUserNamePtr As MQPTR 'Address of LDAP user name'
    LDAPUserNameOffset As Long 'Offset of LDAP user name from start
                                of MQAIR structure'
    LDAPUserNameLength As Long 'Length of LDAP user name'
    LDAPPASSWORD As String*32 'Password to access LDAP server'
End Type
```

### **StrucId (MQCHAR4) pour MQAIR**

Il s'agit de l'identificateur de structure de la structure d'enregistrement des informations d'authentification. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQAIR\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUC\_MQAIR\_**

Identificateur de l'enregistrement d'informations d'authentification.

Pour le langage de programmation C, la constante MQAIR\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQAIR\_STRUC\_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

### **Version (MQLONG) pour MQAIR**

Il s'agit du numéro de version de la structure d'enregistrement des informations d'authentification. Il s'agit toujours d'une zone d'entrée.

Il doit s'agir de l'une des valeurs suivantes :

#### **MQAIR\_VERSION\_1**

Enregistrement des informations d'authentification Version-1 .

Il s'agit de la valeur initiale de cette zone.

#### **MQAIR\_VERSION\_2**

Enregistrement des informations d'authentification Version-2 .

La constante suivante indique le numéro de version de la version en cours:

#### **MQAIR\_CURRENT\_VERSION**

Version actuelle de l'enregistrement des informations d'authentification.

### **AuthInfoType (MQLONG) pour MQAIR**

Il s'agit du type d'informations d'authentification contenues dans l'enregistrement.

La valeur peut être l'un des deux paramètres suivants:

#### **MQAIT\_CRL\_LDAP**

Vérification de la révocation de certificat à l'aide du serveur LDAP.

#### **MQAIT\_OCSP**

Vérification de la révocation de certificat à l'aide d'OCSP.

Si la valeur n'est pas valide, l'appel échoue avec le code anomalie MQRC\_AUTH\_INFO\_TYPE\_ERROR.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est MQAIT\_CRL\_LDAP.

### **AuthInfoConnName (MQCHAR264) pour MQAIR**

Il s'agit du nom d'hôte ou de l'adresse réseau d'un hôte sur lequel le serveur LDAP s'exécute. Il peut être suivi d'un numéro de port facultatif, entre parenthèses. Le numéro de port par défaut est 389.

Si la valeur est inférieure à la longueur de la zone, terminez la valeur par un caractère NULL ou remplissez la zone avec des blancs. Si la valeur n'est pas valide, l'appel échoue avec le code anomalie MQRC\_AUTH\_INFO\_CONN\_NAME\_ERROR.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par MQ\_AUTH\_INFO\_CONN\_NAME\_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.

### **LDAPUserNamePtr (PMQCHAR) pour MQAIR**

Il s'agit du nom d'utilisateur LDAP.

Il se compose du nom distinctif de l'utilisateur qui tente d'accéder au serveur CRL LDAP. Si la valeur est inférieure à la longueur spécifiée par *LDAPUserNameLength*, terminez la valeur par un caractère null ou remplissez la valeur avec des blancs pour la longueur *LDAPUserNameLength*. La zone est ignorée si *LDAPUserNameLength* a la valeur zéro.

Vous pouvez fournir le nom d'utilisateur LDAP de l'une des deux manières suivantes:

- A l'aide de la zone de pointeur *LDAPUserNamePtr*

Dans ce cas, l'application peut déclarer une chaîne distincte de la structure MQAIR et définir *LDAPUserNamePtr* sur l'adresse de la chaîne.

Envisagez d'utiliser *LDAPUserNamePtr* pour les langages de programmation qui prennent en charge le type de données de pointeur d'une manière portable dans différents environnements (par exemple, le langage de programmation C).

- En utilisant la zone de décalage *LDAPUserNameOffset*



Dans ce cas, l'application doit déclarer une structure composée contenant la structure MQSCO suivie du tableau d'enregistrements MQAIR suivi des chaînes de nom d'utilisateur LDAP et définir *LDAPUserNameOffset* sur le décalage de la chaîne de nom appropriée à partir du début de la structure MQAIR. Vérifiez que cette valeur est correcte et qu'elle peut être prise en compte dans un MQLONG (le langage de programmation le plus restrictif est COBOL, pour lequel la plage valide est comprise entre -999 999 999 et +999 999 999).

Envisagez d'utiliser *LDAPUserNameOffset* pour les langages de programmation qui ne prennent pas en charge le type de données de pointeur ou qui implémentent le type de données de pointeur d'une manière qui peut ne pas être portable dans des environnements différents (par exemple, le langage de programmation COBOL).

Quelle que soit la technique choisie, n'utilisez qu'une seule des méthodes *LDAPUserNamePtr* et *LDAPUserNameOffset* ; l'appel échoue avec le code anomalie MQRC\_LDAP\_USER\_NAME\_ERROR si les deux sont différents de zéro.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire.

**Remarque :** Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

### ***LDAPUserNameDécalage (MQLONG) pour MQAIR***

Il s'agit du décalage en octets du nom d'utilisateur LDAP à partir du début de la structure MQAIR.

Le décalage peut être positif ou négatif. La zone est ignorée si *LDAPUserNameLength* a la valeur zéro.

Vous pouvez utiliser *LDAPUserNamePtr* ou *LDAPUserNameOffset* pour spécifier le nom d'utilisateur LDAP, mais pas les deux ; voir la description de la zone *LDAPUserNamePtr* pour plus de détails.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

### ***LDAPUserNameLongueur (MQLONG) pour MQAIR***

Il s'agit de la longueur en octets du nom d'utilisateur LDAP indiqué par la zone *LDAPUserNamePtr* ou *LDAPUserNameOffset*.

La valeur doit être comprise entre zéro et MQ\_DISTINGUISHED\_NAME\_LENGTH. Si la valeur n'est pas valide, l'appel échoue avec le code anomalie MQRC\_LDAP\_USER\_NAME\_LENGTH\_ERR.

Si le serveur LDAP impliqué ne requiert pas de nom d'utilisateur, définissez cette zone sur zéro.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

### ***LDAPPassword (MQCHAR32) pour MQAIR***

Il s'agit du mot de passe requis pour accéder au serveur CRL LDAP. Si la valeur est inférieure à la longueur de la zone, terminez la valeur par un caractère NULL ou remplissez la zone avec des blancs.

Si le serveur LDAP ne requiert pas de mot de passe ou si vous omettez le nom d'utilisateur LDAP, *LDAPPassword* doit être null ou vide. Si vous omettez le nom d'utilisateur LDAP et que *LDAPPassword* n'est pas null ou vide, l'appel échoue avec le code anomalie MQRC\_LDAP\_PASSWORD\_ERROR.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par MQ\_LDAP\_PASSWORD\_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.

### ***OCSPResponderURL (MQCHAR256) pour MQAIR***

Pour une structure MQAIR qui représente les détails de connexion d'un répondeur OCSP, cette zone contient l'URL à laquelle le répondeur peut être contacté.

La valeur de cette zone est une URL HTTP. Cette zone est prioritaire sur une URL dans une extension de certificat AuthorityInfoAccess (AIA).

La valeur est ignorée sauf si les deux instructions suivantes sont vraies:

- La structure MQAIR est à la version 2 ou ultérieure (la zone Version est définie sur MQAIR\_VERSION\_2 ou une version ultérieure).
- La zone AuthInfoType est définie sur MQAIT\_OCSP.

Si la zone ne contient pas d'URL HTTP URL au format correct (et qu'elle n'est pas ignorée), l'appel MQCONNX échoue avec le code anomalie MQRC\_OCSP\_URL\_ERROR.

Cette zone est sensible à la casse. Il doit commencer par la chaîne http:// en minuscules. Le reste de l'URL peut être sensible à la casse, en fonction de l'implémentation du serveur OCSP.

Cette zone n'est pas soumise à la conversion de données.

## MQBMHO-Options de traitement de la mémoire tampon vers le message

La structure MQBMHO permet aux applications de spécifier des options qui contrôlent la façon dont les descripteurs de message sont produits à partir des mémoires tampon. La structure est un paramètre d'entrée dans l'appel MQBUFMH.

### Jeu de caractères et codage

Les données de MQBMHO doivent être dans le jeu de caractères de l'application et dans le codage de l'application (MQENC\_NATIVE).

### Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Tableau 468. Zones dans MQBMHO		
Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID_STRUC_MQBMHO_	'BMHO'
<u>Version</u> (numéro de version de la structure)	MQBMHO_VERSION_1	1
<u>Options</u> (options contrôlant l'action de MQBMHO)	MQBMHO_AUCUN	0
<p><b>Remarques :</b></p> <p>1. Dans le langage de programmation C, la variable macroMQBMHO_DEFAULT contient les valeurs répertoriées dans la table. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</p> <pre>MQBMHO MyBMHO = {MQBMHO_DEFAULT};</pre>		

### Déclarations de langue

Déclaration C pour MQBMHO

```
typedef struct tagMQBMHO MQBMHO;
struct tagMQBMHO {
    MQCHAR4  StrucId;          /* Structure identifier */
    MQLONG   Version;         /* Structure version number */
    MQLONG   Options;         /* Options that control the action of
                               MQBUFMH */
};
```

## Déclaration COBOL pour MQBMHO

```
** MQBMHO structure
10 MQBMHO.
** Structure identifier
15 MQBMHO-STRUCID PIC X(4).
** Structure version number
15 MQBMHO-VERSION PIC S9(9) BINARY.
** Options that control the action of MQBUFMH
15 MQBMHO-OPTIONS PIC S9(9) BINARY.
```

## Déclaration PL/I pour MQBMHO

```
Dcl
1 MQBMHO based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 Options fixed bin(31), /* Options that control the action
of MQBUFMH */
```

## Déclaration High Level Assembler pour MQBMHO

```
MQBMHO DSECT
MQBMHO_STRUCID DS CL4 Structure identifier
MQBMHO_VERSION DS F Structure version number
MQBMHO_OPTIONS DS F Options that control the
* action of MQBUFMH
MQBMHO_LENGTH EQU *-MQBMHO
MQBMHO_AREA DS CL(MQBMHO_LENGTH)
```

### **StrucId (MQCHAR4) pour MQBMHO pour MQBMHO**

Il s'agit de l'identificateur de structure de la mémoire tampon pour la structure de descripteur de message. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQBMHO\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUC\_MQBMHO\_**

Identificateur de la mémoire tampon pour la structure de descripteur de message.

Pour le langage de programmation C, la constante MQBMHO\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQBMHO\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQBMHO pour MQBMHO**

Il s'agit du numéro de version de la structure de la mémoire tampon vers le descripteur de message. Il s'agit toujours d'une zone d'entrée.

La valeur doit être:

#### **MQBMHO\_VERSION\_1**

Numéro de version de la mémoire tampon pour la structure de descripteur de message.

La constante suivante indique le numéro de version de la version en cours:

#### **MQBMHO\_VERSION**

Version en cours de la structure de mémoire tampon à descripteur de message.

### **Options (MQLONG) pour MQBMHO**

Structure de la mémoire tampon vers le descripteur de message-Zone Options

La valeur peut être :

#### **MQBMHO\_DELETE\_PROPERTIES**

Les propriétés ajoutées au descripteur de message sont supprimées de la mémoire tampon. Si l'appel échoue, aucune propriété n'est supprimée.

Options par défaut: si vous n'avez pas besoin de l'option décrite, utilisez l'option suivante:

### **MQBMHO\_AUCUN**

Aucune option spécifiée.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQBMHO\_DELETE\_PROPERTIES.

## **V 9.3.0 MQBNO-Options d'équilibrage**

Le tableau suivant récapitule les zones de la structure.

### **Zones**

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID_STRUC_MQBNO	'BNO↵'
<u>Version</u> (numéro de version de la structure)	MQBNO_VERSION_1	1
<u>ApplicationType</u> (type d'option d'équilibrage défini dans la structure)	MQBNO_VALTYPE_SIMPLE	0
<u>Délai d'attente</u> (délai après lequel le rééquilibrage peut interrompre l'activité de l'application)	MQBNO_TIMEOUT_AS_DEFAULT	0
<u>BalanceOptions</u> (options d'équilibrage définies par l'application émettrice)	MQBNO_OPTIONS_NONE	0

**Remarques :**

1. Le symbole ↵ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable de macro MQBNO\_DEFAULT contient les valeurs répertoriées dans le tableau. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQBNO MyBNO = {MQBNO_DEFAULT};
```

### **Déclarations de langue**

Déclaration C pour MQBNO

```
typedef struct tagMQBNO MQBNO;
struct tagMQBNO {
    MQCHAR4    StructId;           /* Structure identifier */
    MQLONG     Version;           /* Structure version number */
    MQLONG     Type;             /* Type of balancing options set in the
                                structure */
    MQLONG     Timeout;          /* Timeout after which re-balancing might
                                interrupt application activity */
    MQLONG     BalanceOptions;    /* Balancing options set by the issuing
                                application */
};
```

Déclaration COBOL pour MQBNO

```
** MQBNO structure
```

```

10 MQBNO.
**  Structure identifier
   15 MQBNO-STRUCID      PIC X(4).
**  Structure version number
   15 MQBNO-VERSION     PIC S9(9) BINARY.
**  Type of balancing options set in the structure
   15 MQBNO-TYPE        PIC S9(9) BINARY.
**  Timeout after which re-balancing might interrupt application activity
   15 MQBNO-TIMEOUT     PIC S9(9) BINARY.
**  Balancing options set by the issuing application
   15 MQBNO-BALANCEOPTIONS PIC S9(9) BINARY.

```

## Déclaration PL/I pour MQBNO

```

dcl
  1 MQBNO based,
  3 StrucId      char(4),      /* Structure identifier */
  3 Version      fixed bin(31), /* Structure version number */
  3 Type         fixed bin(31), /* Type of balancing options set in the
                               structure*/
  3 Timeout      fixed bin(31), /* Timeout after which re-balancing might
                               interrupt application activity */
  3 BalanceOptions fixed bin(31), /* Balancing options set by the issuing
                               application*/

```

## Référence associée

«MQCNO-Options de connexion», à la page 325

La structure MQCNO permet à l'application de spécifier des options relatives à la connexion au gestionnaire de files d'attente. La structure est un paramètre d'entrée-sortie sur l'appel MQCONN.

### **V 9.3.0** *StrucId (MQCHAR4) pour MQBNO*

Il s'agit de l'identificateur de structure de la structure d'options d'équilibrage. Il s'agit toujours d'une zone d'entrée. Sa valeur initiale est BNO.

La valeur doit être:

#### **BNO**

Identificateur de la structure des options d'équilibrage.

Pour le langage de programmation C, la constante MQBNO\_STRUC\_ID\_ARRAY est également définie. Cette constante a la même valeur que BNO, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Vous devez fournir une valeur valide pour **StrucId** ou MQRC\_BNO\_ERROR est renvoyé.

### **V 9.3.0** *Version (MQLONG) pour MQBNO*

Il s'agit du numéro de version de la structure des options d'équilibrage. Il s'agit toujours d'une zone d'entrée.

La valeur doit être:

#### **MQBNO\_VERSION\_1**

Numéro de version de la structure des options d'équilibrage.

Vous devez fournir une valeur valide pour **Version** ou MQRC\_BNO\_ERROR est renvoyé.

### **V 9.3.0** *ApplicationType (MQLONG) pour MQBNO*

Type d'option d'équilibrage défini dans la structure.

Les valeurs possibles sont les suivantes:

#### **MQBNO\_BALTYPE\_SIMPLE**

Équilibrage simple ; aucune règle spécifique n'est appliquée en plus de celles décrites dans [Influencer le rééquilibrage des applications dans des clusters uniformes](#).

## **MQBNO\_BALTYPE\_REQREP**

Équilibrage de demande-réponse ; après chaque appel MQPUT, un appel MQGET correspondant est attendu pour un message de réponse. L'équilibrage est différé jusqu'à ce qu'un tel message soit reçu ou que le message de demande EXPIRATION ait été dépassé.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQBNO\_BALTYPE\_SIMPLE.

Vous devez indiquer une seule valeur pour la zone **ApplicationType** ou MQRC\_BNO\_ERROR est renvoyé.

**Remarque :** Une valeur supplémentaire pour cette zone de MQBNO\_BALTYPE\_RA\_MANAGED est réservée à l'utilisation par l'adaptateur de ressources IBM MQ pour les environnements JEE. Bien qu'il s'agisse d'une erreur pour une application de fournir cette valeur directement, elle peut, par exemple, être signalée lors de l'interrogation du statut de l'application.

## **V 9.3.0** *Délai d'attente (MQLONG) pour MQBNO*

Le **Timeout** après lequel le rééquilibrage peut interrompre l'activité de l'application.

Les valeurs possibles sont les suivantes:

### **MQBNO\_TIMEOUT\_AS\_DEFAULT**

Valeur de délai d'attente par défaut définie.

### **MQBNO\_TIMEOUT\_IMMEDIATE**

Le délai d'attente immédiat est dépassé.

### **MQBNO\_TIMEOUT\_NEVER**

Aucun délai d'attente n'est dépassé.

La valeur initiale de cette zone est MQBNO\_TIMEOUT\_AS\_DEFAULT.

Vous devez fournir une seule valeur à partir des valeurs définies, ou une valeur comprise entre 0 et 999999999 secondes, pour la zone **Timeout** ou MQRC\_BNO\_ERROR est renvoyé.

## **V 9.3.0** *BalanceOptions (MQLONG) pour MQBNO*

Options d'équilibrage définies par l'application émettrice.

Les valeurs possibles sont les suivantes:

### **MQBNO\_OPTIONS\_NONE**

Aucune option n'est définie

### **MQBNO\_OPTIONS\_IGNORE\_TRANS**

La définition de cette option permet de rééquilibrer les applications même si elles se trouvent au milieu d'une transaction.

La valeur initiale de cette zone est MQBNO\_OPTIONS\_NONE.



Vous pouvez fournir n'importe quelle combinaison des valeurs définies à l'aide du caractère ou de la logique de la zone **BalanceOptions** . Toute valeur non valide entraîne le renvoi d'une erreur MQRC\_BNO\_ERROR.

## **MQBO-Options de début**

La structure MQBO permet à l'application de spécifier des options relatives à la création d'une unité de travail. La structure est un paramètre d'entrée-sortie dans l'appel MQBEGIN.

## **Disponibilité**

La structure MQBO est disponible sur les plateformes suivantes:

-  AIX
-  IBM i

-  Linux
-  Windows

La structure MQBO n'est pas disponible pour IBM MQ MQI clients.

## Jeu de caractères et codage

Les données de MQBO doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et le codage du gestionnaire de files d'attente local indiqué par MQENC\_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Tableau 470. Zones de MQBO pour MQBO		
Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID_STRUC_MQBO_	'B0↵↵'
<u>Version</u> (numéro de version de la structure)	MQBO_VERSION_1	1
<u>Options</u> (options qui contrôlent l'action de MQBEGIN)	MQBO_AUCUN	0
<p><b>Remarques :</b></p> <ol style="list-style-type: none"> <li>1. Le symbole ↵ représente un caractère blanc unique.</li> <li>2. Dans le langage de programmation C, la variable macroMQBO_DEFAULT contient les valeurs répertoriées dans la table. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</li> </ol> <pre>MQBO MyBO = {MQBO_DEFAULT};</pre>		

## Déclarations de langue

Déclaration C pour MQBO

```
typedef struct tagMQBO MQBO;
struct tagMQBO {
    MQCHAR4  StrucId; /* Structure identifier */
    MQLONG   Version; /* Structure version number */
    MQLONG   Options; /* Options that control the action of MQBEGIN */
};
```

Déclaration COBOL pour MQBO

```
** MQBO structure
10 MQBO.
** Structure identifier
15 MQBO-STRUCID PIC X(4).
** Structure version number
15 MQBO-VERSION PIC S9(9) BINARY.
** Options that control the action of MQBEGIN
15 MQBO-OPTIONS PIC S9(9) BINARY.
```

## Déclaration PL/I pour MQBO

```
dc1
  1 MQBO based,
  3 StrucId char(4),      /* Structure identifier */
  3 Version fixed bin(31), /* Structure version number */
  3 Options fixed bin(31); /* Options that control the action of
                          MQBEGIN */
```

## Déclaration Visual Basic pour MQBO

```
Type MQBO
  StrucId As String*4 'Structure identifier'
  Version As Long     'Structure version number'
  Options As Long     'Options that control the action of MQBEGIN'
End Type
```

### **StrucId (MQCHAR4) pour MQBO**

Il s'agit de l'identificateur de structure de la structure des options de début. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQBO\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUC\_MQBO\_**

Identificateur de la structure des options de début.

Pour le langage de programmation C, la constante MQBO\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQBO\_STRUC\_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

### **Version (MQLONG) pour MQBO**

Il s'agit du numéro de version de la structure des options de début. Il s'agit toujours d'une zone d'entrée.

La valeur doit être:

#### **MQBO\_VERSION\_1**

Numéro de version de la structure des options de début.

La constante suivante indique le numéro de version de la version en cours:

#### **MQBO\_CURRENT\_VERSION**

Version actuelle de la structure des options de début.

### **Options (MQLONG) pour MQBO**

Cette zone est toujours une zone d'entrée. Sa valeur initiale est MQBO\_NONE.

La valeur doit être:

#### **MQBO\_AUCUN**

Aucune option spécifiée.

## **MQCBC-Contexte de rappel**

La structure MQCBC permet de spécifier les informations de contexte transmises à une fonction de rappel. La structure est un paramètre d'entrée-sortie dans l'appel à une routine de consommateur de message.

## **Disponibilité**

La structure MQCBC est disponible sur les plateformes suivantes:

-  AIX
-  IBM i



-  Linux
-  Windows
-  z/OS

et pour IBM MQ MQI clients connecté à ces systèmes.

## Version

La version actuelle de MQCBC est MQCBC\_VERSION\_2.

## Jeu de caractères et codage

Les données de MQCBC doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par MQENC\_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure sera dans le jeu de caractères et le codage du client.

## Zones

Il n'existe aucune valeur initiale pour la structure **MQCBC**. La structure est transmise en tant que paramètre à une routine de rappel. Le gestionnaire de files d'attente initialise la structure ; les applications ne l'initialisent jamais.

### Remarques :

- Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.
- Il n'existe aucune valeur initiale pour la structure MQCBC. La structure est transmise en tant que paramètre à une routine de rappel. Le gestionnaire de files d'attente initialise la structure ; les applications ne l'initialisent jamais.

<i>Tableau 471. Zones dans MQCBC</i>	
<b>Zone</b>	<b>Description</b>
<u>StrucID</u>	Identificateur de structure
<u>Version</u>	Numéro de version de structure
<u>CallType</u>	Pourquoi la fonction a été appelée
<u>HOBJ</u>	Descripteur d'objet
<u>CallbackArea</u>	Zone de la fonction de rappel à utiliser
<u>ConnectionArea</u>	Zone de la fonction de rappel à utiliser
<u>CompCode</u>	Code de fin d'exécution
<u>Motif</u>	Code raison
<u>Etat</u>	Indication de l'état du consommateur actuel
<u>DataLength</u>	LONGUEUR DE MESSAGE
<u>BufferLength</u>	Longueur de la mémoire tampon de message en octets
<u>Indicateurs</u>	Indicateurs généraux
<b>Remarque :</b> La zone restante est ignorée si la version est inférieure à MQCBC_VERSION_2	
<u>ReconnectDelay</u>	Nombre de millisecondes avant la tentative de reconnexion

## Déclarations de langue

### Déclaration C pour MQCBC

```
typedef struct tagMQCBC MQCBC;
struct tagMQCBC {
    MQCHAR4   StructId;           /* Structure identifier */
    MQLONG    Version;           /* Structure version number */
    MQLONG    CallType;         /* Why Function was called */
    MQHOBj    Hobj;             /* Object Handle */
    MQPTR     CallbackArea;     /* Callback data passed to the function */
    MQPTR     ConnectionArea;   /* MQCTL data area passed to the function */
    MQLONG    CompCode;        /* Completion Code */
    MQLONG    Reason;          /* Reason Code */
    MQLONG    State;          /* Consumer State */
    MQLONG    DataLength;      /* Message Data Length */
    MQLONG    BufferLength;     /* Buffer Length */
    MQLONG    Flags;          /* Flags containing information about
                               this consumer */

    /* Ver:1 */
    MQLONG    ReconnectDelay;   /* Number of milliseconds before */
    /* Ver:2 */ };             /* reconnect attempt */
```

### Déclaration COBOL pour MQCBC

```
** MQCBC structure
10 MQCBC.
** Structure Identifier
15 MQCBC-STRUCID PIC X(4).
** Structure Version
15 MQCBC-VERSION PIC S9(9) BINARY.
** Call Type
15 MQCBC-CALLTYPE PIC S9(9) BINARY.
** Object Handle
15 MQCBC-HOBJ PIC S9(9) BINARY.
** Callback User Area
15 MQCBC-CALLBACKAREA POINTER
** Connection Area
15 MQCBC-CONNECTIONAREA POINTER
** Completion Code
15 MQCBC-COMPCODE PIC S9(9) BINARY.
** Reason Code
15 MQCBC-REASON PIC S9(9) BINARY.
** Consumer State
15 MQCBC-STATE PIC S9(9) BINARY.
** Data Length
15 MQCBC-DATALENGTH PIC S9(9) BINARY.
** Buffer Length
15 MQCBC-BUFFERLENGTH PIC S9(9) BINARY.
** Flags
15 MQCBC-FLAGS PIC S9(9) BINARY.
** Ver:1 **
** Number of milliseconds before reconnect attempt
15 MQCBC-RECONNECTDELAY PIC S9(9) BINARY.
** Ver:2 **
```

### Déclaration PL/I pour MQCBC

```
dcl
1 MQCBC based,
3 StructId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version */
3 CallType fixed bin(31), /* Callback type */
3 Hobj fixed bin(31), /* Object Handle */
3 CallbackArea pointer, /* User area passed to the function */
3 ConnectionArea pointer, /* Connection User Area */
3 CompCode fixed bin(31); /* Completion Code */
3 Reason fixed bin(31); /* Reason Code */
3 State fixed bin(31); /* Consumer State */
3 DataLength fixed bin(31); /* Message Data Length */
3 BufferLength fixed bin(31); /* Message Buffer length */
3 Flags fixed bin(31); /* Consumer Flags */
/* Ver:1 */
```

```

3 ReconnectDelay      fixed bin(31); /* Number of milliseconds before */
/* Ver:2 */           /* reconnect attempt */

```

## Déclaration High Level Assembler pour MQCBC

```

MQCBC                DSECT
MQCBC                DS  0F    Force fullword alignment
MQCBC_STRUCID        DS  CL4   Structure identifier
MQCBC_VERSION        DS  F     Structure version number
MQCBC_CALLTYPE       DS  F     Why Function was called
MQCBC_HOBJ           DS  F     Object Handle
MQCBC_CALLBACKAREA   DS  A     Callback data passed to the function
MQCBC_CONNECTIONAREA DS  A     MQCTL Data area passed to the function
MQCBC_COMPCODE       DS  F     Completion Code
MQCBC_REASON         DS  F     Reason Code
MQCBC_STATE          DS  F     Consumer State
MQCBC_DATALENGTH     DS  F     Message Data Length
MQCBC_BUFFERLENGTH   DS  F     Buffer Length
MQCBC_FLAGS          DS  F     Flags containing information about this consumer
MQCBC_RECONNECTDELAY DS  F     Number of milliseconds before reconnect
MQCBC_LENGTH         EQU *-MQCBC
                    ORG      MQCBC
MQCBC_AREA           DS  CL(MQCBC_LENGTH)

```

### **StrucId (MQCHAR4) pour MQCBC**

Il s'agit de l'identificateur de structure de la structure de contexte de rappel. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQCBC\_STRUC\_ID.

La valeur doit être:

#### **MQCBC\_STRUC\_ID**

Identificateur de la structure de contexte de rappel.

Pour le langage de programmation C, la constante MQCBC\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQCBC\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQCBC**

Il s'agit du numéro de version de la structure de contexte de rappel. Il s'agit toujours d'une zone d'entrée.

La valeur doit être:

#### **MQCBC\_VERSION\_1**

Structure de contexte de rappel version 1.

La constante suivante indique le numéro de version de la version en cours:

#### **MQCBC\_CURRENT\_VERSION**

Version actuelle de la structure de contexte de rappel.

La fonction de rappel est toujours transmise à la dernière version de la structure.

### **CallType (MQLONG) pour MQCBC**

Zone contenant des informations sur la raison pour laquelle cette fonction a été appelée ; les valeurs suivantes sont définies.

Types d'appel de distribution de message: ces types d'appel contiennent des informations sur un message. Les paramètres **DataLength** et **BufferLength** sont valides pour ces types d'appel.

#### **MQCBCT\_MSG\_REMOVED**

La fonction de consommateur de message a été appelée avec un message qui a été supprimé de façon destructive de l'identificateur d'objet.

Si la valeur de *CompCode* est MQCC\_WARNING, la valeur de la zone *Reason* est MQRC\_TRUNCATED\_MSG\_ACCEPTED ou l'un des codes indiquant un problème de conversion de données.

## **MQCBCT\_MSG\_NOT\_REMOVED**

La fonction de consommateur de message a été appelée avec un message qui n'a pas encore été supprimé de façon destructive de l'identificateur d'objet. Le message peut être supprimé de façon destructive de l'identificateur d'objet à l'aide de *MsgToken*.

Le message n'a peut-être pas été supprimé pour les raisons suivantes:

- Les options MQGMO ont demandé une opération de navigation, MQGMO\_BROWSE\_\*
- Le message est plus grand que la mémoire tampon disponible et les options MQGMO ne spécifient pas MQGMO\_ACCEPT\_TRUNCATED\_MSG

Si la valeur de *CompCode* est MQCC\_WARNING, la valeur de la zone *Reason* est MQRC\_TRUNCATED\_MSG\_FAILED ou l'un des codes indiquant un problème de conversion de données.

Types d'appel de contrôle de rappel: ces types d'appel contiennent des informations sur le contrôle du rappel et ne contiennent pas de détails sur un message. Ces types d'appel sont demandés à l'aide des [options](#) de la structure MQCBD.

Les paramètres **DataLength** et **BufferLength** ne sont pas valides pour ces types d'appel.

## **APPEL MQCBCT\_REGISTER\_CALL**

Le but de ce type d'appel est de permettre à la fonction de rappel d'effectuer une configuration initiale.

La fonction de rappel est appelée immédiatement après l'enregistrement du rappel, c'est-à-dire lors du retour d'un appel MQCB à l'aide d'une valeur pour la zone *Operation* de MQOP\_REGISTER.

Ce type d'appel est utilisé à la fois pour les consommateurs de messages et les gestionnaires d'événements.

S'il est demandé, il s'agit du premier appel de la fonction de rappel.

La valeur de la zone *Reason* est MQRC\_NONE.

## **Appel MQCBCT\_START\_CALL**

Le but de ce type d'appel est de permettre à la fonction de rappel d'effectuer une configuration lors de son démarrage, par exemple, en réinstanciant des ressources qui ont été nettoyées lors de son arrêt précédent.

La fonction de rappel est appelée lorsque la connexion est démarrée à l'aide de MQOP\_START ou de MQOP\_START\_WAIT.

Si une fonction de rappel est enregistrée dans une autre fonction de rappel, ce type d'appel est appelé lorsque le rappel est renvoyé.

Ce type d'appel est utilisé uniquement pour les consommateurs de message.

La valeur de la zone *Reason* est MQRC\_NONE.

## **MQCBCT\_STOP\_CALL**

Le but de ce type d'appel est de permettre à la fonction de rappel d'effectuer un nettoyage lorsqu'elle est arrêtée pendant un certain temps, par exemple, en nettoyant des ressources supplémentaires qui ont été acquises lors de la consommation de messages.

La fonction de rappel est appelée lorsqu'un appel MQCTL est émis à l'aide d'une valeur pour la zone *Operation* de MQOP\_STOP.

Ce type d'appel est utilisé uniquement pour les consommateurs de message.

La valeur de la zone *Reason* est définie pour indiquer la raison de l'arrêt.

## **MQCBCT\_DEREGISTER\_CALL**

Le but de ce type d'appel est de permettre à la fonction de rappel d'effectuer un nettoyage final à la fin du processus de consommation. La fonction de rappel est appelée lorsque:

- La fonction de rappel est désenregistrée à l'aide d'un appel MQCB avec MQOP\_REGISTER\_.

- La file d'attente est fermée, ce qui entraîne un désenregistrement implicite. Dans cette instance, la fonction de rappel est transmise à MQHO\_UNUSABLE\_HOBY en tant que descripteur d'objet.
- L'appel MQDISC se termine-provoquant une fermeture implicite et, par conséquent, un désenregistrement. Dans ce cas, la connexion n'est pas immédiatement déconnectée et toute transaction en cours n'est pas encore validée.

Si l'une de ces actions est effectuée dans la fonction de rappel elle-même, l'action est appelée une fois que le rappel est renvoyé.

Ce type d'appel est utilisé à la fois pour les consommateurs de messages et les gestionnaires d'événements.

S'il est demandé, il s'agit du dernier appel de la fonction de rappel.

La valeur de la zone *Reason* est définie pour indiquer la raison de l'arrêt.

## **MQCBCT\_EVENT\_CALL**

### **Fonction du gestionnaire d'événements**

La fonction de gestionnaire d'événements a été appelée sans message lorsque le gestionnaire de files d'attente ou la connexion s'arrête ou se met au repos.

Cet appel peut être utilisé pour effectuer l'action appropriée pour toutes les fonctions de rappel.

### **Fonction de consommateur de message**

La fonction de consommateur de message a été appelée sans message lorsqu'une erreur (*CompCode* = MQCC\_FAILED) spécifique au descripteur d'objet a été détectée ; par exemple, *Reason code* = MQRC\_GET\_INHIBÉ.

La valeur de la zone *Reason* est définie pour indiquer la raison de l'appel.

## **MQCBCT\_MC\_EVENT\_CALL**

La fonction de gestionnaire d'événements a été appelée pour les événements de multidiffusion. Le gestionnaire d'événements reçoit IBM MQ des événements de multidiffusion au lieu d'événements 'normaux' IBM MQ .

Pour plus d'informations sur MQCBCT\_MC\_EVENT\_CALL, voir [Génération de rapports sur les exceptions de multidiffusion](#).

## **Hobj (MQHOBY) pour MQCBC**

Il s'agit du descripteur d'objet pour les appels au consommateur de message.

Pour un gestionnaire d'événements, cette valeur est MQHO\_NONE

L'application peut utiliser ce descripteur et le jeton de message dans le bloc Options d'extraction de message pour extraire le message si un message n'a pas été supprimé de la file d'attente.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQHO\_UNUSABLE\_HOBY

## **CallbackArea (MQPTR) pour MQCBC**

Cette zone est disponible pour la fonction de rappel à utiliser.

Le gestionnaire de files d'attente ne prend aucune décision en fonction du contenu de cette zone et est transmis tel quel à partir de la zone [CallbackArea](#) de la structure MQCBD, qui est un paramètre de l'appel MQCB utilisé pour définir la fonction de rappel.

Les modifications apportées à *CallbackArea* sont conservées dans les appels de la fonction de rappel pour un *Hobj*. Cette zone n'est pas partagée avec les fonctions de rappel pour les autres descripteurs.

Il s'agit d'un champ d'entrée/sortie de la fonction de rappel. La valeur initiale de cette zone est un pointeur NULL ou des octets NULL.

### **ConnectionArea (MQPTR) pour MQCBC**

Cette zone est disponible pour la fonction de rappel à utiliser.

Le gestionnaire de files d'attente ne prend aucune décision en fonction du contenu de cette zone et est transmis tel quel à partir de la zone ConnectionArea de la structure MQCTLO, qui est un paramètre de l'appel MQCTL utilisé pour contrôler la fonction de rappel.

Toutes les modifications apportées à cette zone par les fonctions de rappel sont conservées dans les appels de la fonction de rappel. Cette zone peut être utilisée pour transmettre des informations qui doivent être partagées par toutes les fonctions de rappel. Contrairement à *CallbackArea*, cette zone est commune à tous les rappels d'un descripteur de connexion.

Il s'agit d'une zone d'entrée et de sortie. La valeur initiale de cette zone est un pointeur NULL ou des octets NULL.

### **CompCode (MQLONG) pour MQCBC**

Cette zone indique le code achèvement. Indique s'il y a eu des problèmes lors de la consommation du message.

La valeur est l'une des suivantes :

#### **MQCC\_OK**

Achèvement réussi

#### **MQCC\_WARNING**

Avertissement (achèvement partiel)

#### **MQCC\_FAILED**

Echec de l'appel

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est MQCC\_OK.

### **Motif (MQLONG) pour MQCBC**

Il s'agit du code raison qualifiant le *CompCode*.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est MQRC\_NONE.

### **Etat (MQLONG) pour MQCBC**

Indication de l'état du consommateur en cours. Cette zone est la plus importante pour une application lorsqu'un code raison différent de zéro est transmis à la fonction de consommateur.

Vous pouvez utiliser cette zone pour simplifier la programmation d'application car vous n'avez pas besoin de coder le comportement de chaque code anomalie.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est MQCS\_NONE

Tableau 472.		
Etat	Action du gestionnaire de files d'attente	Valeur de la constante
MQCS_NONE Ce code anomalie représente un appel normal sans informations supplémentaires sur la raison	Non ; il s'agit du fonctionnement normal.	0

Tableau 472. (suite)		
Etat	Action du gestionnaire de files d'attente	Valeur de la constante
<i>MQCS_SUSPENDED_TEMPORARY</i> Ces codes anomalie représentent des conditions temporaires.	La routine de rappel est appelée pour signaler la condition, puis suspendue. Au bout d'un certain temps, le système peut tenter à nouveau l'opération, ce qui peut entraîner une nouvelle levée de la même condition.	1
<i>MQCS_SUSPENDED_USER_ACTION</i> Ces codes anomalie représentent les conditions dans lesquelles le rappel doit intervenir pour résoudre la condition.	Le consommateur est suspendu et la routine de rappel est appelée pour signaler la condition. La routine de rappel doit résoudre la condition si possible et soit RESUME, soit fermer la connexion.	2
<i>MQCS_SUSPENDED</i> Ces codes anomalie représentent des échecs qui empêchent d'autres rappels de message.	Le gestionnaire de files d'attente interrompt automatiquement la fonction de rappel. Si la fonction de rappel est reprise, il est probable qu'elle reçoive à nouveau le même code anomalie.	3
<i>MQCS_STOPPED</i> Ces codes anomalie représentent la fin de la consommation des messages.	Distribué au gestionnaire d'exceptions et aux rappels qui ont spécifié MQCBDO_STOP_CALL. Aucun autre message ne peut être consommé.	4

### **DataLength (MQLONG) pour MQCBC**

Il s'agit de la longueur en octets des données d'application du message. Si la valeur est zéro, cela signifie que le message ne contient pas de données d'application.

La zone DataLength contient la longueur du message, mais pas nécessairement la longueur des données de message transmises au consommateur. Il se peut que le message ait été tronqué. Utilisez la zone ReturnedLength dans le MQGMO pour déterminer la quantité de données qui a été réellement transmise au consommateur.

Si le code anomalie indique que le message a été tronqué, vous pouvez utiliser la zone DataLength pour déterminer la taille réelle du message. Cela vous permet de déterminer la taille de la mémoire tampon requise pour accueillir les données de message, puis d'émettre un appel MQCB pour mettre à jour MaxMsgLength avec une valeur appropriée.

Si l'option MQGMO\_CONVERT est spécifiée, le message converti peut être supérieur à la valeur renvoyée pour DataLength. Dans de tels cas, l'application doit probablement émettre un appel MQCB pour mettre à jour MaxMsgLength pour qu'il soit supérieur à la valeur renvoyée par le gestionnaire de files d'attente pour DataLength.

Pour éviter les problèmes de troncature de message, spécifiez la longueur MaxMsgs sous la forme MQCBD\_FULL\_MSG\_LENGTH. Ainsi, le gestionnaire de files d'attente alloue une mémoire tampon pour la longueur complète des messages après la conversion des données. Sachez toutefois que même si cette option est spécifiée, il est possible que l'espace de stockage disponible soit insuffisant pour traiter correctement la demande. Les applications doivent toujours vérifier le code anomalie renvoyé. Par exemple, s'il n'est pas possible d'allouer suffisamment de mémoire pour convertir le message, les messages sont renvoyés à l'application non convertie.

Il s'agit d'une zone d'entrée de la fonction de consommateur de message ; elle n'est pas pertinente pour une fonction de gestionnaire d'événements.

### **BufferLength (MQLONG) pour MQCBC**

Cette zone indique la longueur, en octets, de la mémoire tampon de messages qui a été transmise à cette fonction.

La mémoire tampon peut être supérieure à la valeur MaxMsgLength définie pour le consommateur et à la valeur ReturnedLength dans le MQGMO.

La longueur réelle des messages est indiquée dans la zone [DataLength](#) .

L'application peut utiliser la totalité de la mémoire tampon à ses propres fins pendant la durée de la fonction de rappel.

Il s'agit d'une zone d'entrée de la fonction de consommateur de message ; elle n'est pas pertinente pour une fonction de gestionnaire d'exceptions.

### **Indicateurs (MQLONG) pour MQCBC**

Indicateurs contenant des informations sur ce consommateur.

L'option suivante est définie:

#### **MQCBCF\_READA\_BUFFER\_EMPTY**

Cet indicateur peut être renvoyé si un appel MQCLOSE précédent utilisant l'option MQCO QUIESCE a échoué avec le code anomalie MQRC\_READ\_AHEAD\_MSGS.

Ce code indique que le dernier message de lecture anticipée est renvoyé et que la mémoire tampon est à présent vide. Si l'application émet un autre appel MQCLOSE à l'aide de l'option MQCO QUIESCE), elle aboutit.

Notez qu'il n'est pas garanti qu'une application reçoit un message avec cet indicateur défini, car il peut encore y avoir des messages dans la mémoire tampon de lecture anticipée qui ne correspondent pas aux critères de sélection en cours. Dans cette instance, la fonction de consommateur est appelée avec le code anomalie MQRC\_HOBJ QUIESCED.

Si la mémoire tampon de lecture anticipée est complètement vide, le consommateur est appelé avec l'indicateur MQCBCF\_READA\_BUFFER\_EMPTY et le code anomalie MQRC\_HOBJ QUIESCED\_NO\_MSGS.

Il s'agit d'une zone d'entrée de la fonction de consommateur de message ; elle n'est pas pertinente pour une fonction de gestionnaire d'événements.

### **ReconnectDelay (MQLONG) pour MQCBC**

ReconnectDelay indique la durée pendant laquelle le gestionnaire de files d'attente attend avant de tenter de se reconnecter. La zone peut être modifiée par un gestionnaire d'événements pour modifier complètement le délai ou arrêter la reconnexion.

Utilisez la zone ReconnectDelay uniquement si la valeur de la zone Reason dans le contexte de rappel est MQRC\_RECONNECTING.

Lors de l'entrée dans le gestionnaire d'événements, la valeur de ReconnectDelay correspond au nombre de millisecondes pendant lesquelles le gestionnaire de files d'attente va attendre avant d'effectuer une tentative de reconnexion. Le [Tableau 473](#), à la page 296 répertorie les valeurs que vous pouvez définir pour modifier le comportement du gestionnaire de files d'attente en cas de retour du gestionnaire d'événements.

Nom	Valeur	Description
MQRD_NO_RECONNECT	-1	Ne faites plus de tentatives de reconnexion. Une erreur est renvoyée à l'application.
MQRD_NO_DELAY	0	Essayez de vous reconnecter immédiatement.



Tableau 473. Valeurs de ReconnectDelay (suite)

Nom	Valeur	Description
Milliseconds	>0	Attendez ce nombre de millisecondes avant de retenter la connexion.

## MQCBD-Descripteur de rappel

La structure MQCBD est utilisée pour spécifier une fonction de rappel et les options contrôlant son utilisation par le gestionnaire de files d'attente. La structure est un paramètre d'entrée dans l'appel MQCB.

### Disponibilité

La structure MQCBD est disponible sur les plateformes suivantes:

-  AIX
-  IBM i
-  Linux
-  Windows
-  z/OS

et pour IBM MQ MQI clients connecté à ces systèmes.

### Version

La version actuelle de MQCBD est MQCBD\_VERSION\_1.

### Jeu de caractères et codage

Les données dans MQCBD doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et le codage du gestionnaire de files d'attente local indiqué par MQENC\_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

### Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Tableau 474. Zones dans MQCBD

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucID</u> (identificateur de structure)	ID MQCBD_STRUC_ID	'CBD↵'
<u>Version</u> (numéro de version de la structure)	MQCBD_VERSION_1	1
<u>CallbackType</u> (type de fonction de rappel)	MQCBT_MESSAGE_CONSUMER	1
<u>Options</u> (options contrôlant la consommation des messages)	MQCBDO_AUCUN	0

Tableau 474. Zones dans MQCBD (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>CallbackArea</u> (zone à utiliser pour la fonction de rappel)	Aucun	Pointeur null ou blancs null
<u>CallbackFunction</u> (si la fonction est appelée en tant qu'appel d'API)	Aucun	Pointeur null ou blancs null
<u>CallbackName</u> (si la fonction est appelée en tant que programme lié dynamiquement)	Aucun	Chaîne nulle ou blancs
<u>MaxMsgLongueur</u> (longueur du message le plus long pouvant être lu)	MQCBD_LONGUEUR_MS G_COMPLET	-1
<p><b>Remarques :</b></p> <ol style="list-style-type: none"> <li>1. Le symbole ~ représente un caractère blanc unique.</li> <li>2. La valeur "chaîne nulle" ou "blancs" indique la chaîne nulle dans le langage de programmation C, et les caractères blancs dans les autres langages de programmation.</li> <li>3. Dans le langage de programmation C, la variable macroMQCBD_DEFAULT contient les valeurs répertoriées dans la table. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</li> </ol> <pre>MQCBD MyCBD = {MQCBD_DEFAULT};</pre>		

## Déclarations de langue

### Déclaration C pour MQCBD

```
typedef struct tagMQCBD MQCBD;
struct tagMQCBD {
    MCHAR4      StrucId;           /* Structure identifier */
    MQLONG      Version;          /* Structure version number */
    MQLONG      CallBackType;     /* Callback function type */
    MQLONG      Options;          /* Options controlling message
    consumption */
    MQPTR       CallbackArea;     /* User data passed to the function */
    MQPTR       CallbackFunction; /* Callback function pointer */
    MCHAR128    CallbackName;     /* Callback name */
    MQLONG      MaxMsgLength;     /* Maximum message length */
};
```

### Déclaration COBOL pour MQCBD

```
** MQCBD structure
10 MQCBD.
** Structure Identifier
15 MQCBD-STRUCID          PIC X(4).
** Structure Version
15 MQCBD-VERSION        PIC S9(9) BINARY.
** Callback Type
15 MQCBD-CALLBACKTYPE   PIC S9(9) BINARY.
** Options
15 MQCBD-OPTIONS        PIC S9(9) BINARY.
** Callback User Area
15 MQCBD-CALLBACKAREA   POINTER
** Callback Function Pointer
15 MQCBD-CALLBACKFUNCTION FUNCTION-POINTER
** Callback Program Name
15 MQCBD-CALLBACKNAME   PIC X(128)
```

```
** Maximum Message Length
15 MQCDB-MAXMSGLENGTH PIC S9(9) BINARY.
```

## Déclaration PL/I pour MQCBD

```
dcl
1 MQCBD based,
3 StrucId char(4), /* Structure identifier*/
3 Version fixed bin(31), /* Structure version*/
3 CallbackType fixed bin(31), /* Callback function type */
3 Options fixed bin(31), /* Options */
3 CallbackArea pointer, /* User area passed to the function */
3 CallbackFunction pointer, /* Callback Function Pointer */
3 CallbackName char(128), /* Callback Program Name */
3 MaxMsgLength fixed bin(31); /* Maximum Message Length */
```

### **StrucId (MQCHAR4) pour MQCBD**

Il s'agit de l'identificateur de structure de la structure de descripteur de rappel. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQCBD\_STRUC\_ID.

La valeur doit être:

#### **ID MQCBD\_STRUC\_ID**

Identificateur de la structure de descripteur de rappel.

Pour le langage de programmation C, la constante MQCBD\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQCBD\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQCBD**

Il s'agit du numéro de version de la structure de descripteur de rappel. Il s'agit toujours d'une zone d'entrée.

La valeur doit être:

#### **MQCBD\_VERSION\_1**

Structure de descripteur de rappel version 1.

La constante suivante indique le numéro de version de la version en cours:

#### **MQCBD\_CURRENT\_VERSION**

Version actuelle de la structure de descripteur de rappel.

### **CallbackType (MQLONG) pour MQCBD**

Structure de descripteur de rappel-Zone CallbackType

Il s'agit du type de la fonction de rappel. Les valeurs suivantes sont possibles :

#### **MQCBT\_MESSAGE\_CONSUMER**

Définit ce rappel en tant que fonction de consommateur de message.

Une fonction de rappel de consommateur de message est appelée lorsqu'un message répondant aux critères de sélection spécifiés est disponible sur un descripteur d'objet et que la connexion est démarrée.

#### **MQCBT\_EVENT\_HANDLER**

Définit ce rappel comme la routine d'événement asynchrone ; il n'est pas piloté pour consommer des messages pour un descripteur.

*Hobj* n'est pas requis sur l'appel MQCB définissant le gestionnaire d'événements et est ignoré s'il est spécifié.

Le gestionnaire d'événements est appelé pour les conditions qui affectent l'ensemble de l'environnement du consommateur de messages. La fonction de consommateur est appelée sans message lorsqu'un événement, par exemple l'arrêt ou la mise au repos d'un gestionnaire de files

d'attente ou d'une connexion, se produit. Elle n'est pas appelée pour les conditions spécifiques à un seul consommateur de message, par exemple, MQRC\_GET\_INHIBÉ.

Les événements sont distribués à l'application, que la connexion soit démarrée ou arrêtée, sauf dans les environnements suivants:

- Environnement CICS on z/OS
- applications sans unités d'exécution

Si l'appelant ne transmet pas l'une de ces valeurs, l'appel échoue avec le code *Reason* MQRC\_CALLBACK\_TYPE\_ERROR

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQCBT\_MESSAGE\_CONSUMER.

### **Options (MQLONG) pour MQCBD**

Structure de descripteur de rappel-Zone Options

Vous pouvez spécifier une ou plusieurs de ces options. Pour spécifier plusieurs options, ajoutez les valeurs ensemble (n'ajoutez pas la même constante plusieurs fois) ou combinez les valeurs à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations bit).

#### **MQCBDO\_ECHEC\_IF QUIESCING**

L'appel MQCB échoue si le gestionnaire de files d'attente est à l'état de mise au repos.

Sous z/OS, cette option force également l'échec de l'appel MQCB si la connexion (pour une application CICS ou IMS ) est à l'état de mise au repos.

Spécifiez MQGMO\_FAIL\_IF QUIESCING, dans les options MQGMO transmises à l'appel MQCB, pour envoyer une notification aux consommateurs de message lorsqu'ils sont mis au repos.

**Options de contrôle:** Les options suivantes contrôlent si la fonction de rappel est appelée, sans message, lorsque l'état du consommateur change:

#### **APPEL MQCBDO\_REGISTER\_CALL**

La fonction de rappel est appelée avec le type d'appel MQCBCT\_REGISTER\_CALL.

#### **Appel MQCBDO\_START\_CALL**

La fonction de rappel est appelée avec le type d'appel MQCBCT\_START\_CALL.

#### **Appel MQCBDO\_STOP\_CALL**

La fonction de rappel est appelée avec le type d'appel MQCBCT\_STOP\_CALL.

#### **MQCBDO\_DEREGISTER\_CALL**

La fonction de rappel est appelée avec le type d'appel MQCBCT\_DEREGISTER\_CALL.

#### **MQCBDO\_EVENT\_CALL**

La fonction de rappel est appelée avec le type d'appel MQCBCT\_EVENT\_CALL.

#### **MQCBDO\_MC\_EVENT\_CALL**

La fonction de rappel est appelée avec le type d'appel MQCBCT\_MC\_EVENT\_CALL.

Voir [CallType](#) pour plus de détails sur ces types d'appel.

**Option par défaut:** si vous n'avez besoin d'aucune des options décrites, utilisez l'option suivante:

#### **MQCBDO\_AUCUN**

Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. Toutes les options sont définies sur leur valeur par défaut.

MQCBDO\_NONE est défini pour faciliter la documentation du programme ; il n'est pas prévu que cette option soit utilisée avec d'autres, mais comme sa valeur est zéro, une telle utilisation ne peut pas être détectée.

Il s'agit d'une zone d'entrée. La valeur initiale de la zone *Options* est MQCBDO\_NONE.

### **CallbackArea (MQPTR) pour MQCBD**

Structure de descripteur de rappel-Zone CallbackArea

Il s'agit d'une zone disponible que la fonction de rappel peut utiliser.

Le gestionnaire de files d'attente ne prend aucune décision en fonction du contenu de cette zone et est transmis tel quel à partir de la zone `CallbackArea` de la structure `MQCBC`, qui est un paramètre de la déclaration de fonction de rappel.

La valeur est utilisée uniquement sur un *Operation* ayant une valeur `MQOP_REGISTER`, sans rappel actuellement défini, elle ne remplace pas une définition précédente.

Il s'agit d'une zone d'entrée et de sortie de la fonction de rappel. La valeur initiale de cette zone est un pointeur `NULL` ou des octets `NULL`.

### ***CallbackFunction (MQPTR) pour MQCBD***

Structure de descripteur de rappel-Zone `CallbackFunction`


La fonction de rappel est appelée en tant qu'appel de fonction.

Utilisez cette zone pour spécifier un pointeur vers la fonction de rappel.

Vous devez spécifier *CallbackFunction* ou *CallbackName*. Si vous spécifiez les deux, le code anomalie `MQRC_CALLBACK_ROUTINE_ERROR` est renvoyé.

Si ni *CallbackName* ni *CallbackFunction* n'est défini, l'appel échoue avec le code anomalie `MQRC_CALLBACK_ROUTINE_ERROR`.

Cette option n'est pas prise en charge dans l'environnement suivant: Langages de programmation et compilateurs qui ne prennent pas en charge les références de pointeur de fonction. Dans de telles situations, l'appel échoue avec le code anomalie `MQRC_CALLBACK_ROUTINE_ERROR`.

 Sous `z/OS`, la fonction doit être appelée avec les conventions de liaison du système d'exploitation. Par exemple, dans le langage de programmation C, spécifiez:

```
#pragma linkage(MQCB_FUNCTION,OS)
```

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est un pointeur `NULL` ou des octets `NULL`.

**Remarque :** Lors de l'utilisation de CICS avec IBM WebSphere MQ 7.0.1, la consommation asynchrone est prise en charge si:

- Apar PK66866 est appliqué à CICS TS 3.2
- Apar PK89844 est appliqué à CICS TS 4.1

### ***CallbackName (MQCHAR128) pour MQCBD***

Structure de descripteur de rappel-Zone `CallbackName`

La fonction de rappel est appelée en tant que programme lié dynamiquement.

Vous devez spécifier *CallbackFunction* ou *CallbackName*. Si vous spécifiez les deux, le code anomalie `MQRC_CALLBACK_ROUTINE_ERROR` est renvoyé.

Si *CallbackName* ou *CallbackFunction* n'est pas défini, l'appel échoue avec le code anomalie `MQRC_CALLBACK_ROUTINE_ERROR`.

Le module est chargé lorsque la première routine de rappel à utiliser est enregistrée et déchargé lorsque la dernière routine de rappel à utiliser désenregistre.

Sauf indication contraire dans le texte suivant, le nom est cadré à gauche dans la zone, sans espaces imbriqués ; le nom lui-même est complété par des blancs à la longueur de la zone. Dans les descriptions qui suivent, les crochets ([ ]) indiquent des informations facultatives:

#### **IBM i**

Le nom de rappel peut être l'un des formats suivants:

- Programme de bibliothèque `"/`
- Bibliothèque `"/` ServiceProgram ("`FunctionName`")

Par exemple, `MyLibrary/MyProgram(MyFunction)`.

Le nom de la bibliothèque peut être \*LIBL. Les noms de bibliothèque et de programme sont limités à un maximum de 10 caractères.

### **AIX and Linux**

Le nom de rappel est le nom d'un module ou d'une bibliothèque chargeable dynamiquement, suffixé avec le nom d'une fonction résidant dans cette bibliothèque. Le nom de la fonction doit être placé entre parenthèses. Le nom de la bibliothèque peut éventuellement être précédé d'un chemin de répertoire:

```
[path]library(function)
```

Si le chemin n'est pas spécifié, le chemin de recherche système est utilisé.

Le nom est limité à un maximum de 128 caractères.

### **Windows**

Le nom de rappel est le nom d'une bibliothèque de liens dynamiques, suffixé avec le nom d'une fonction résidant dans cette bibliothèque. Le nom de la fonction doit être placé entre parenthèses. Le nom de la bibliothèque peut éventuellement être précédé d'un chemin de répertoire et d'une unité:

```
[d:][path]library(function)
```

Si l'unité et le chemin ne sont pas spécifiés, le chemin de recherche système est utilisé.

Le nom est limité à un maximum de 128 caractères.

### **z/OS**

Le nom de rappel est le nom d'un module de chargement qui est valide pour la spécification sur le paramètre EP de la macro LINK ou LOAD.

Le nom est limité à un maximum de 8 caractères.

### **z/OS CICS**

Le nom de rappel est le nom d'un module de chargement valide pour la spécification sur le paramètre PROGRAM de la macro de la commande EXEC CICS LINK.

Le nom est limité à un maximum de 8 caractères.

Le programme peut être défini comme programme distant à l'aide de l'option REMOTESYSTEM de la définition PROGRAM installée ou par le programme de routage dynamique.

La région CICS distante doit être connectée à IBM MQ si le programme doit utiliser des appels API IBM MQ. Notez toutefois que la zone [Hobj](#) de la structure MQCBC n'est pas valide sur un système distant.

Si un incident se produit lors de la tentative de chargement de *CallbackName*, l'un des codes d'erreur suivants est renvoyé à l'application:

- MQRC\_MODULE\_NOT\_FOUND
- MQRC\_MODULE\_INVALID
- MQRC\_MODULE\_ENTRY\_NOT\_FOUND

Un message est également consigné dans le journal des erreurs contenant le nom du module pour lequel le chargement a été tenté et le code anomalie du système d'exploitation.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est une chaîne nulle ou des blancs.

### **MaxMsgLongueur (MQLONG) pour MQCBD**

Il s'agit de la longueur en octets du message le plus long qui peut être lu à partir du descripteur et transmis à la routine de rappel. Structure de descripteur de rappel- MaxMsgZone Longueur

Si un message a une longueur plus longue, la routine de rappel reçoit *MaxMsgLength* octets du message et le code anomalie suivant:

- MQRC\_TRUNCATED\_MSG\_FAILED ou
- MQRC\_TRUNCATED\_MSG\_ACCEPTED si vous avez spécifié MQGMO\_ACCEPT\_TRUNCATED\_MSG.

La longueur réelle des messages est indiquée dans la zone `DataLength` de la structure MQCBC.

La valeur spéciale suivante est définie:

#### **MQCBD\_LONGUEUR\_MSG\_COMPLET**

La longueur de la mémoire tampon est ajustée par le système pour renvoyer les messages sans troncature.

Si la mémoire disponible est insuffisante pour allouer une mémoire tampon pour recevoir le message, le système appelle la fonction de rappel avec un code anomalie MQRC\_STORAGE\_NOT\_AVAILABLE.

Si, par exemple, vous demandez une conversion de données et que la mémoire disponible est insuffisante pour convertir les données de message, le message non converti est transmis à la fonction de rappel.

Il s'agit d'une zone d'entrée. La valeur initiale de la zone `MaxMsgLength` est MQCBD\_FULL\_MSG\_LENGTH.

## **MQCHARV-Chaîne de longueur variable**

Utilisez la structure MQCHARV pour décrire une chaîne de longueur variable.

### **Disponibilité**

La structure MQCHARV est disponible sur les plateformes suivantes:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour IBM MQ MQI clients connecté à ces systèmes.

### **Jeu de caractères et codage**

Les données du MQCHARV doivent être dans le codage du gestionnaire de files d'attente local fourni par MQENC\_NATIVE et dans le jeu de caractères de la zone VSCCSID de la structure. Si l'application s'exécute en tant que client MQ, la structure doit être dans le codage du client. Certains jeux de caractères ont une représentation qui dépend du codage. Si VSCCSID est l'un de ces jeux de caractères, le codage utilisé est le même que celui des autres zones du MQCHARV. Le jeu de caractères identifié par VSCCSID peut être un jeu de caractères codé sur deux octets (DBCS).

### **Utilisation**

La structure MQCHARV traite les données qui peuvent être discontinuës à la structure qui les contient. Pour traiter ces données, les zones déclarées avec le type de données de pointeur peuvent être utilisées. Notez que COBOL ne prend pas en charge le type de données de pointeur dans tous les environnements. Pour cette raison, les données peuvent également être traitées à l'aide de zones qui contiennent le décalage des données à partir du début de la structure contenant le MQCHARV.

### **Zones**

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Tableau 475. Zones dans MQCHARV

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>VSPtr</u> (pointeur vers la chaîne de longueur variable)	Aucun	Pointeur NULL ou octets NULL.
<u>VSOffset</u> (décalage en octets de la chaîne de longueur variable à partir du début de la structure qui contient cette structure MQCHARV)	Aucun	0
<u>VSBufSize</u> (taille en octets de la mémoire tampon adressée par la zone VSPtr ou VSOffset)	MQVS_USE_VSLENGTH	0
<u>VSLength</u> (longueur en octets de la chaîne de longueur variable adressée par la zone VSPtr ou VSOffset)	Aucun	0
<u>VSCCSID</u> (identificateur de jeu de caractères de la chaîne de longueur variable adressée par la zone VSPtr ou VSOffset)	MQCCSI_APPL	-3
<p><b>Remarque :</b> Dans le langage de programmation C, la variable de macro MQCHARV_DEFAULT contient les valeurs répertoriées dans le tableau. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</p> <pre>MQCHARV MyVarStr = {MQCHARV_DEFAULT};</pre>		

## Déclarations de langue

### Déclaration C pour MQCHARV

```
typedef struct tagMQCHARV MQCHARV;
struct tagMQCHARV {
    MQPTR    VSPtr;                /* Address of variable length string */
    MQLONG   VSOffset;            /* Offset of variable length string */
    MQLONG   VSBufSize;          /* Size of buffer */
    MQLONG   VSLength;           /* Length of variable length string */
    MQLONG   VSCCSID;            /* CCSID of variable length string */
};
```

### Déclaration COBOL pour MQCHARV

```
** MQCHARV structure
10 MQCHARV.
** Address of variable length string
15 MQCHARV-VSPTR    POINTER.
** Offset of variable length string
15 MQCHARV-VSOFFSET PIC S9(9) BINARY.
** Size of buffer
15 MQCHARV-VSBUFSIZE PIC S9(9) BINARY.
** Length of variable length string
15 MQCHARV-VSLENGTH PIC S9(9) BINARY.
** CCSID of variable length string
15 MQCHARV-VSCCSID  PIC S9(9) BINARY.
```

**Remarque :** Si vous souhaitez porter une application COBOL entre des environnements, vous devez déterminer si le type de données de pointeur est disponible dans tous les environnements prévus. Si ce n'est pas le cas, l'application doit traiter les données à l'aide des zones de décalage au lieu des zones de pointeur. Dans les environnements où les pointeurs ne sont pas pris en charge, vous pouvez déclarer les zones de pointeur en tant que chaînes d'octets de la longueur appropriée, la valeur initiale étant la chaîne



d'octets tout-null. Ne modifiez pas cette valeur initiale si vous utilisez les zones de décalage. Pour ce faire, sans modifier les fichiers de stockage fournis, vous pouvez utiliser les éléments suivants:

```
COPY CMQCHRVV REPLACING POINTER BY ==BINARY PIC S9(9)==.
```

où CMQCHRVV peut être échangé pour le copybook à utiliser.

Déclaration PL/I pour MQCHARV

```
dcl
  1 MQCHARV based,
  3 VSPtr      pointer,      /* Address of variable length string */
  3 VSOffset   fixed bin(31), /* Offset of variable length string */
  3 VSBufSize  fixed bin(31), /* Size of buffer */
  3 VSLength   fixed bin(31), /* Length of variable length string */
  3 VSCCSID    fixed bin(31); /* CCSID of variable length string */
```

Déclaration High Level Assembler pour MQCHARV

```
MQCHARV          DSECT
MQCHARV_VSPTR    DS  F      Address of variable length string
MQCHARV_VSOFFSET DS  F      Offset of variable length string
MQCHARV_VSBUFSIZE DS  F      Size of buffer
MQCHARV_VSLENGTH DS  F      Length of variable length string
MQCHARV_VSCCSID DS  F      CCSID of variable length string
*
MQCHARV_LENGTH   EQU  *-MQCHARV
                  ORG  MQCHARV
MQCHARV_AREA     DS      CL(MQCHARV_LENGTH)
```

### ***VSPtr (MQPTR) pour MQCHARV***

Il s'agit d'un pointeur vers la chaîne de longueur variable.

Vous pouvez utiliser la zone VSPtr ou VSOffset pour spécifier la chaîne de longueur variable, mais pas les deux.

La valeur initiale de cette zone est un pointeur NULL ou des octets NULL.

### ***VSOffset (MQLONG) pour MQCHARV***

Le décalage peut être positif ou négatif. Vous pouvez utiliser la zone VSPtr ou VSOffset pour spécifier la chaîne de longueur variable, mais pas les deux. Décalage en octets de la chaîne de longueur variable à partir du début de MQCHARV ou de la structure qui la contient.

Lorsque la structure MQCHARV est imbriquée dans une autre structure, cette valeur correspond au décalage en octets de la chaîne de longueur variable à partir du début de la structure qui contient cette structure MQCHARV. Lorsque la structure MQCHARV n'est pas imbriquée dans une autre structure, par exemple, si elle est spécifiée en tant que paramètre dans un appel de fonction, le décalage est relatif au début de la structure MQCHARV.

La valeur initiale de cette zone est 0.

### ***VSBufSize (MQLONG) pour MQCHARV***

Il s'agit de la taille en octets de la mémoire tampon adressée par la zone VSPtr ou VSOffset.

Lorsque la structure MQCHARV est utilisée comme champ de sortie dans un appel de fonction, cette zone doit être initialisée avec la longueur de la mémoire tampon fournie. Si la valeur de VSLength est supérieure à VSBufSize, seuls VSBufSize octets de données sont renvoyés à l'appelant dans la mémoire tampon.

Cette valeur doit être une valeur supérieure ou égale à zéro, ou la valeur spéciale suivante reconnue:

## **MQVS\_USE\_VSLENGTH**

Lorsqu'elle est spécifiée, la longueur de la mémoire tampon est extraite de la zone VSLength de la structure MQCHARV. N'utilisez pas cette valeur lorsque vous utilisez la structure comme zone de sortie et qu'une mémoire tampon est fournie.

Il s'agit de la valeur initiale de cette zone.

## **VSLength (MQLONG) pour MQCHARV**

Longueur en octets de la chaîne de longueur variable adressée par la zone VSPtr ou VSOOffset.

La valeur initiale de cette zone est 0. La valeur doit être supérieure ou égale à zéro ou la valeur spéciale suivante qui est reconnue:

## **MQVS\_NULL\_TERMINATED**

Si MQVS\_NULL\_TERMINATED n'est pas spécifié, les octets VSLength sont inclus dans la chaîne. Si des caractères nuls sont présents, ils ne délimitent pas la chaîne.

Si MQVS\_NULL\_TERMINATED est spécifié, la chaîne est délimitée par la première valeur null rencontrée dans la chaîne. La valeur null elle-même n'est pas incluse dans cette chaîne.

**Remarque :** Le caractère null utilisé pour mettre fin à une chaîne si MQVS\_NULL\_TERMINATED est spécifié est une valeur null provenant du jeu de codes spécifié par VSCCSID.


Par exemple, dans UTF-16 (CCSID 1200, 13488 et 17584), il s'agit du codage Unicode à deux octets dans lequel une valeur null est représentée par un nombre de 16 bits de tous les zéros. En UTF-16, il est courant de rechercher des octets uniques définis sur zéro qui font partie de caractères (caractères ASCII 7 bits par exemple), mais les chaînes ne se terminent par une valeur nulle que lorsque deux octets 'zéro' sont trouvés sur une limite d'octet pair. Il est possible d'obtenir deux octets 'zéro' sur une limite impaire lorsqu'ils font partie de caractères valides. Par exemple, x'01'x'00 x'00'00'x'30' représente deux caractères Unicode valides et ne termine pas la chaîne.

## **VSCCSID (MQLONG) pour MQCHARV**

Il s'agit de l'identificateur de jeu de caractères de la chaîne de longueur variable adressée par la zone **VSPtr** ou **VSOOffset**.

La valeur initiale de cette zone est *MQCCSI\_APPL*, qui est définie par MQ pour indiquer qu'elle doit être remplacée par l'identificateur de jeu de caractères true du processus en cours. Par conséquent, la valeur de la constante *MQCCSI\_APPL* n'est jamais associée à une chaîne de longueur variable.

La valeur initiale de cette zone peut être modifiée en définissant une valeur différente pour la constante *MQCCSI\_APPL* de votre unité de compilation. La façon dont vous effectuez cette opération dépend du langage de programmation de votre application.

 Sur les systèmes z/OS, l'application par défaut CCSID utilisée par *MQCCSI\_APPL* est définie comme suit:

- Pour les applications par lots LE utilisant l'interface DLL, la valeur par défaut est CODESET associée à l'environnement local en cours lors de l'émission de **MQCONN** (la valeur par défaut est 1047).
- Pour les applications par lots LE liées à l'un des stubs MQ par lots, la valeur par défaut est CODESET associée à l'environnement local en cours au moment du premier appel MQI émis après **MQCONN** (la valeur par défaut est 1047).
- Pour les applications par lots non LE s'exécutant sur une unité d'exécution z/OS UNIX System Services, la valeur par défaut est la valeur de THLICCSID au moment du premier appel MQI émis après **MQCONN** (la valeur par défaut est 1047).
- Pour les autres applications par lots, la valeur par défaut est CCSID du gestionnaire de files d'attente.

## **Redéfinition de MQCCSI\_APPL**

Les exemples suivants montrent comment remplacer la valeur de *MQCCSI\_APPL* dans différents langages de programmation. Vous pouvez modifier la valeur de *MQCCSI\_APPL* en supprimant la nécessité de définir le VSCCSID pour chaque chaîne de longueur variable séparément. Dans ces exemples, le CCSID

est défini sur 1208 ; remplacez-le par la valeur requise. Il s'agit de la valeur par défaut, que vous pouvez remplacer en définissant le VSCCSID dans n'importe quelle instance spécifique de MQCHARV.

Utilisation de C

```
#define MQCCSI_APPL 1208
#include <cmqc.h>
```

Utilisation COBOL

```
COPY CMQXYZV REPLACING -3 BY 1208.
```

Utilisation de PL/I

```
%MQCCSI_APPL = '1208';
%include syslib(cmqp);
```

Utilisation de l' High Level Assembler

```
MQCCSI_APPL EQU 1208
CMQA LIST=NO
```

## MQCIH-en-tête CICS bridge





La structure MQCIH décrit les informations d'en-tête d'un message envoyé à CICS via CICS bridge.

Pour toute plateforme prise en charge par IBM MQ , vous pouvez créer et transmettre un message qui inclut la structure MQCIH, mais seul un gestionnaire de files d'attente IBM MQ for z/OS peut utiliser CICS bridge. Par conséquent, pour que le message puisse être envoyé à CICS à partir d'un gestionnaire de files d'attente nonz/OS , votre réseau de gestionnaires de files d'attente doit inclure au moins un gestionnaire de files d'attente z/OS par lequel le message peut être acheminé.

Toutes les versions de CICS prises en charge par IBM MQ 9.0.0et les versions ultérieures utilisent la version fournie par CICS du pont. Pour plus d'informations sur la configuration de l'adaptateur IBM MQ CICS et des composants de IBM MQ CICS bridge, voir la section [Configuration des connexions à MQ](#) dans la documentation CICS.

## Disponibilité

La structure MQCIH est disponible sur les plateformes suivantes:

-  AIX
-  Linux
-  Windows
-  z/OS

et pour IBM MQ MQI clients connecté à ces systèmes.

## Format

MQFMT\_CICS

## Version

La version actuelle de MQCIH est MQCIH\_VERSION\_2. Les zones qui existent uniquement dans la version la plus récente de la structure sont identifiées comme telles dans les descriptions qui suivent.

Les fichiers d'en-tête, COPY et INCLUDE fournis pour les langages de programmation pris en charge contiennent la version la plus récente de MQCIH, avec la valeur initiale de la zone *Version* définie sur MQCIH\_VERSION\_2.

## Jeu de caractères et codage

Des conditions spéciales s'appliquent au jeu de caractères et au codage utilisés pour la structure MQCIH et les données de message d'application:

- Les applications qui se connectent au gestionnaire de files d'attente qui possède la file d'attente CICS bridge doivent fournir une structure MQCIH qui se trouve dans le jeu de caractères et le codage du gestionnaire de files d'attente. En effet, la conversion de données de la structure MQCIH n'est pas effectuée dans ce cas.
- Les applications qui se connectent à d'autres gestionnaires de files d'attente peuvent fournir une structure MQCIH qui se trouve dans l'un des jeux de caractères et codages pris en charge ; l'agent de canal de réception des messages connecté au gestionnaire de files d'attente qui possède la file d'attente CICS bridge convertit la structure MQCIH.
- Les données de message d'application qui suivent la structure MQCIH doivent être dans le même jeu de caractères et dans le même codage que la structure MQCIH. Vous ne pouvez pas utiliser les zones *CodedCharSetId* et *Encoding* de la structure MQCIH pour spécifier le jeu de caractères et le codage des données de message d'application.

Vous devez fournir un exit de conversion de données pour convertir les données de message d'application si les données ne sont pas l'un des formats intégrés pris en charge par le gestionnaire de files d'attente.

## Utilisation

Si l'application requiert des valeurs identiques aux valeurs initiales affichées dans Tableau 477, à la page 309 et que le pont s'exécute avec AUTH=LOCAL ou AUTH=IDENTIFY, vous pouvez omettre la structure MQCIH du message. Dans tous les autres cas, la structure doit être présente.

Le pont accepte une structure MQCIH version-1 ou version-2, mais pour les transactions 3270, vous devez utiliser une structure version-2.

L'application doit s'assurer que les zones documentées en tant que zones de demande ont des valeurs appropriées dans le message envoyé au pont ; ces zones sont entrées dans le pont.

Les zones documentées en tant que zones de réponse sont définies par le CICS bridge dans le message de réponse que le pont envoie à l'application. Les informations d'erreur sont renvoyées dans les zones *ReturnCode*, *Function*, *CompCode*, *Reason* et *AbendCode*, mais elles ne sont pas toutes définies dans tous les cas. Le tableau suivant indique les zones qui sont définies pour différentes valeurs de *ReturnCode*.

<i>Tableau 476. Contenu des zones d'informations d'erreur dans la structure MQCIH pour MQCIH</i>				
<b>ReturnCode</b>	<b>Function</b>	<b>CompCode</b>	<b>Reason</b>	<b>AbendCode</b>
MQCRC_OK	-	-	-	-
MQCRC_BRIDGE_ERREUR	-	-	MQFB_CICS_*	-
MQCRC_MQ_API_ERREUR MQCRC_BRIDGE_TIMEOUT	Nom d'appel MQ	MQ CompCode	MQ Reason	-
MQCRC_CICS_EXEC_ERROR MQCRC_SECURITY_ERROR MQCRC_PROGRAM_NOT_AVAILABLE MQCRC_TRANSID_NOT_AVAILABLE	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	-
MQCRC_BRIDGE_ABEND MQCRC_APPLICATION_ABEND	-	-	-	CICS ABCODE

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Tableau 477. Zones de MQCIH pour MQCIH

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID_STRUC_MQCIH	'CIH~'
<u>Version</u> (numéro de version de la structure)	MQCIH_VERSION_2	2
<u>StrucLength</u> (longueur de la structure MQCIH)	MQCIH_LENGTH_2	180
<u>Codage</u> (réservé)	Aucun	0
<u>CodedCharSetId</u> (réservé)	Aucun	0
<u>Format</u> (nom de formatMQ des données qui suivent MQCIH)	MQFMT_AUCUN	Espaces vides
<u>Indicateurs</u> (indicateurs)	MQCIH_NONE	0
<u>ReturnCode</u> (code retour du pont)	MQCRC_OK	0
<u>CompCode</u> (code achèvementMQ ou CICS EIBRESP)	MQCC_OK	0
<u>Raison</u> (MQ raison ou code retour, ou CICS EIBRESP2)	MQRC_NONE	0
<u>UOWControl</u> (contrôle d'unité de travail)	MQCUOWC_ONLY	273
<u>GetWaitIntervalle</u> (intervalle d'attente pour l'appel MQGET émis par la tâche de pont)	MQCGWI_VALEUR PAR DEFAULT	-2
<u>LinkType</u> (type de lien)	PROGRAMME MQCL	1
<u>OutputDataLongueur</u> (longueur des données COMMAREA de sortie)	MQCODL_AS_ENTREE	-1
<u>FacilityKeepTime</u> (heure de libération de la fonction de pont)	Aucun	0
<u>ADSDescriptor</u> (descripteur ADS d'envoi/de réception)	MQCADSD_NONE	0
<u>ConversationalTask</u> (indique si la tâche peut être conversationnelle)	MQCCT_NO	0
<u>StatutTaskEnd</u> (statut à la fin de la tâche)	MQCTES_NOSYNC	0
<u>Facility</u> (jeton de fonction de pont)	MQCFAC_NONE	Valeurs NULL
<u>Fonction</u> (nom d'appelMQ ou fonction CICS EIBFN)	MQCFUNC_NONE	Espaces vides
<u>AbendCode</u> (code de fin anormale)	Aucun	Espaces vides
<u>Authentificateur</u> (mot de passe ou mot de passe)	Aucun	Espaces vides
<u>Reserved1</u> (réservé)	Aucun	Espaces vides
<u>ReplyToReplyTo</u> (nom de formatMQ du message de réponse)	MQFMT_AUCUN	Espaces vides
<u>RemoteSysId</u> (ID système CICS distant à utiliser)	Aucun	Espaces vides

Tableau 477. Zones de MQCIH pour MQCIH (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
IDRemoteTrans (CICS RTRANSID à utiliser)	Aucun	Espaces vides
TransactionId (transaction à joindre)	Aucun	Espaces vides
FacilityLike (attributs émulsés de terminal)	Aucun	Espaces vides
AttentionId (touche AID)	Aucun	Espaces vides
StartCode (code de début de transaction)	MQCSC_NONE	Espaces vides
CancelCode (code de transaction de fin anormale)	Aucun	Espaces vides
NextTransactionId (transaction suivante à associer)	Aucun	Espaces vides
Reserved2 (réservé)	Aucun	Espaces vides
Reserved3 (réservé)	Aucun	Espaces vides
<b>Remarque :</b> Les zones restantes ne sont pas présentes si <i>Version</i> est inférieur à MQCIH_VERSION_2.		
CursorPosition (position du curseur)	Aucun	0
ErrorOffset (décalage de l'erreur dans le message)	Aucun	0
InputItem (élément d'entrée)	Aucun	0
Reserved4 (réservé)	Aucun	0
<p><b>Remarques :</b></p> <ol style="list-style-type: none"> <li>Le symbole ~ représente un caractère blanc unique.</li> <li>Dans le langage de programmation C, la variable macroMQCIH_DEFAULT contient les valeurs répertoriées dans la table. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</li> </ol> <pre style="background-color: #f0f0f0; padding: 10px;">MQCIH MyCIH = {MQCIH_DEFAULT};</pre>		

## Déclarations de langue

Déclaration C pour MQCIH

```
typedef struct tagMQCIH MQCIH;
struct tagMQCIH {
    MQCHAR4  StrucId;           /* Structure identifier */
    MQLONG   Version;          /* Structure version number */
    MQLONG   StrucLength;      /* Length of MQCIH structure */
    MQLONG   Encoding;         /* Reserved */
    MQLONG   CodedCharSetId;   /* Reserved */
    MQCHAR8  Format;           /* MQ format name of data that follows
                               MQCIH */
    MQLONG   Flags;            /* Flags */
    MQLONG   ReturnCode;       /* Return code from bridge */
    MQLONG   CompCode;         /* MQ completion code or CICS EIBRESP */
    MQLONG   Reason;           /* MQ reason or feedback code, or CICS
                               EIBRESP2 */
    MQLONG   UOWControl;       /* Unit-of-work control */
    MQLONG   GetWaitInterval;  /* Wait interval for MQGET call issued
                               by bridge task */
    MQLONG   LinkType;         /* Link type */
    MQLONG   OutputDataLength; /* Output COMMAREA data length */
};
```

```

MQLONG FacilityKeepTime; /* Bridge facility release time */
MQLONG ADSDescriptor; /* Send/receive ADS descriptor */
MQLONG ConversationalTask; /* Whether task can be conversational */
MQLONG TaskEndStatus; /* Status at end of task */
MQBYTE8 Facility; /* Bridge facility token */
MQCHAR4 Function; /* MQ call name or CICS EIBFN
function */

MQCHAR4 AbendCode; /* Abend code */
MQCHAR8 Authenticator; /* Password or passticket */
MQCHAR8 Reserved1; /* Reserved */
MQCHAR8 ReplyToFormat; /* MQ format name of reply message */
MQCHAR4 RemoteSysId; /* Reserved */
MQCHAR4 RemoteTransId; /* Reserved */
MQCHAR4 TransactionId; /* Transaction to attach */
MQCHAR4 FacilityLike; /* Terminal emulated attributes */
MQCHAR4 AttentionId; /* AID key */
MQCHAR4 StartCode; /* Transaction start code */
MQCHAR4 CancelCode; /* Abend transaction code */
MQCHAR4 NextTransactionId; /* Next transaction to attach */
MQCHAR8 Reserved2; /* Reserved */
MQCHAR8 Reserved3; /* Reserved */
MQLONG CursorPosition; /* Cursor position */
MQLONG ErrorOffset; /* Offset of error in message */
MQLONG InputItem; /* Reserved */
MQLONG Reserved4; /* Reserved */
};

```

## Déclaration COBOL pour MQCIH

```

** MQCIH structure
10 MQCIH.
** Structure identifier
15 MQCIH-STRUCID PIC X(4).
** Structure version number
15 MQCIH-VERSION PIC S9(9) BINARY.
** Length of MQCIH structure
15 MQCIH-STRUCLength PIC S9(9) BINARY.
** Reserved
15 MQCIH-ENCODING PIC S9(9) BINARY.
** Reserved
15 MQCIH-CODEDCHARSETID PIC S9(9) BINARY.
** MQ format name of data that follows MQCIH
15 MQCIH-FORMAT PIC X(8).
** Flags
15 MQCIH-FLAGS PIC S9(9) BINARY.
** Return code from bridge
15 MQCIH-RETURNCode PIC S9(9) BINARY.
** MQ completion code or CICS EIBRESP
15 MQCIH-COMPCODE PIC S9(9) BINARY.
** MQ reason or feedback code, or CICS EIBRESP2
15 MQCIH-REASON PIC S9(9) BINARY.
** Unit-of-work control
15 MQCIH-UOWCONTROL PIC S9(9) BINARY.
** Wait interval for MQGET call issued by bridge task
15 MQCIH-GETWAITINTERVAL PIC S9(9) BINARY.
** Link type
15 MQCIH-LINKTYPE PIC S9(9) BINARY.
** Output COMMAREA data length
15 MQCIH-OUTPUTDATALENGTH PIC S9(9) BINARY.
** Bridge facility release time
15 MQCIH-FACILITYKEEPTime PIC S9(9) BINARY.
** Send/receive ADS descriptor
15 MQCIH-ADSDESCRIPTOR PIC S9(9) BINARY.
** Whether task can be conversational
15 MQCIH-CONVERSATIONALTASK PIC S9(9) BINARY.
** Status at end of task
15 MQCIH-TASKENDSTATUS PIC S9(9) BINARY.
** Bridge facility token
15 MQCIH-FACILITY PIC X(8).
** MQ call name or CICS EIBFN function
15 MQCIH-FUNCTION PIC X(4).
** Abend code
15 MQCIH-ABENDCODE PIC X(4).
** Password or passticket
15 MQCIH-AUTHENTICATOR PIC X(8).
** Reserved
15 MQCIH-RESERVED1 PIC X(8).
** MQ format name of reply message
15 MQCIH-REPLYTOFORMAT PIC X(8).

```

```

**      Reserved
15 MQCIH-REMOTESYSID      PIC X(4).
**      Reserved
15 MQCIH-REMOTETRANSID   PIC X(4).
**      Transaction to attach
15 MQCIH-TRANSACTIONID   PIC X(4).
**      Terminal emulated attributes
15 MQCIH-FACILITYLIKE    PIC X(4).
**      AID key
15 MQCIH-ATTENTIONID     PIC X(4).
**      Transaction start code
15 MQCIH-STARTCODE       PIC X(4).
**      Abend transaction code
15 MQCIH-CANCELCODE      PIC X(4).
**      Next transaction to attach
15 MQCIH-NEXTTRANSACTIONID PIC X(4).
**      Reserved
15 MQCIH-RESERVED2       PIC X(8).
**      Reserved
15 MQCIH-RESERVED3       PIC X(8).
**      Cursor position
15 MQCIH-CURSORPOSITION  PIC S9(9) BINARY.
**      Offset of error in message
15 MQCIH-ERROROFFSET     PIC S9(9) BINARY.
**      Reserved
15 MQCIH-INPUTITEM       PIC S9(9) BINARY.
**      Reserved
15 MQCIH-RESERVED4       PIC S9(9) BINARY.

```

### Déclaration PL/I pour MQCIH

```

dcl
  1 MQCIH based,
  3 StrucId          char(4),          /* Structure identifier */
  3 Version          fixed bin(31),   /* Structure version number */
  3 StrucLength      fixed bin(31),   /* Length of MQCIH structure */
  3 Encoding         fixed bin(31),   /* Reserved */
  3 CodedCharSetId   fixed bin(31),   /* Reserved */
  3 Format            char(8),          /* MQ format name of data that
                                     follows MQCIH */
  3 Flags            fixed bin(31),   /* Flags */
  3 ReturnCode       fixed bin(31),   /* Return code from bridge */
  3 CompCode         fixed bin(31),   /* MQ completion code or CICS
                                     EIBRESP */
  3 Reason           fixed bin(31),   /* MQ reason or feedback code, or
                                     CICS EIBRESP2 */
  3 UOWControl       fixed bin(31),   /* Unit-of-work control */
  3 GetWaitInterval fixed bin(31),   /* Wait interval for MQGET call
                                     issued by bridge task */
  3 LinkType         fixed bin(31),   /* Link type */
  3 OutputDataLength fixed bin(31),   /* Output COMMAREA data length */
  3 FacilityKeepTime fixed bin(31),   /* Bridge facility release time */
  3 ADSDescriptor    fixed bin(31),   /* Send/receive ADS descriptor */
  3 ConversationalTask fixed bin(31), /* Whether task can be
                                     conversational */
  3 TaskEndStatus    fixed bin(31),   /* Status at end of task */
  3 Facility         char(8),          /* Bridge facility token */
  3 Function         char(4),          /* MQ call name or CICS EIBFN
                                     function */
  3 AbendCode       char(4),          /* Abend code */
  3 Authenticator    char(8),          /* Password or passticket */
  3 Reserved1       char(8),          /* Reserved */
  3 ReplyToFormat    char(8),          /* MQ format name of reply
                                     message */
  3 RemoteSysId     char(4),          /* Reserved */
  3 RemoteTransId   char(4),          /* Reserved */
  3 TransactionId    char(4),          /* Transaction to attach */
  3 FacilityLike     char(4),          /* Terminal emulated attributes */
  3 AttentionId     char(4),          /* AID key */
  3 StartCode       char(4),          /* Transaction start code */
  3 CancelCode      char(4),          /* Abend transaction code */
  3 NextTransactionId char(4),        /* Next transaction to attach */
  3 Reserved2       char(8),          /* Reserved */
  3 Reserved3       char(8),          /* Reserved */
  3 CursorPosition  fixed bin(31),   /* Cursor position */
  3 ErrorOffset     fixed bin(31),   /* Offset of error in message */
  3 InputItem       fixed bin(31),   /* Reserved */
  3 Reserved4       fixed bin(31);   /* Reserved */

```



## Déclaration High Level Assembler pour MQCIH

```

MQCIH                DSECT
MQCIH_STRUCID        DS    CL4  Structure identifier
MQCIH_VERSION        DS    F    Structure version number
MQCIH_STRUCLNGTH     DS    F    Length of MQCIH structure
MQCIH_ENCODING       DS    F    Reserved
MQCIH_CODEDCHARSETID DS    F    Reserved
MQCIH_FORMAT         DS    CL8  MQ format name of data that follows
*
MQCIH_FLAGS          DS    F    Flags
MQCIH_RETURNCODE     DS    F    Return code from bridge
MQCIH_COMPCODE       DS    F    MQ completion code or CICS EIBRESP
MQCIH_REASON         DS    F    MQ reason or feedback code, or CICS
*
MQCIH_UOWCONTROL     DS    F    Unit-of-work control
MQCIH_GETWAITINTERVAL DS    F    Wait interval for MQGET call issued
*
MQCIH_LINKTYPE       DS    F    Link type
MQCIH_OUTPUTDATALENGTH DS    F    Output COMMAREA data length
MQCIH_FACILITYKEEPTIME DS    F    Bridge facility release time
MQCIH_ADSDESCRIPTOR  DS    F    Send/receive ADS descriptor
MQCIH_CONVERSATIONALTASK DS    F    Whether task can be conversational
MQCIH_TASKENDSTATUS  DS    F    Status at end of task
MQCIH_FACILITY       DS    XL8  Bridge facility token
MQCIH_FUNCTION       DS    CL4  MQ call name or CICS EIBFN function
MQCIH_ABENDCODE      DS    CL4  Abend code
MQCIH_AUTHENTICATOR  DS    CL8  Password or passticket
MQCIH_RESERVED1     DS    CL8  Reserved
MQCIH_REPLYTOFORMAT  DS    CL8  MQ format name of reply message
MQCIH_REMOTESYSID    DS    CL4  Reserved
MQCIH_REMOTETRANSID  DS    CL4  Reserved
MQCIH_TRANSACTIONID  DS    CL4  Transaction to attach
MQCIH_FACILITYLIKE   DS    CL4  Terminal emulated attributes
MQCIH_ATTENTIONID    DS    CL4  AID key
MQCIH_STARTCODE      DS    CL4  Transaction start code
MQCIH_CANCELCODE     DS    CL4  Abend transaction code
MQCIH_NEXTTRANSACTIONID DS    CL4  Next transaction to attach
MQCIH_RESERVED2     DS    CL8  Reserved
MQCIH_RESERVED3     DS    CL8  Reserved
MQCIH_CURSORPOSITION DS    F    Cursor position
MQCIH_ERROROFFSET    DS    F    Offset of error in message
MQCIH_INPUTITEM      DS    F    Reserved
MQCIH_RESERVED4     DS    F    Reserved
*
MQCIH_LENGTH         EQU    *-MQCIH
*
MQCIH_AREA           ORG    MQCIH
                     DS    CL(MQCIH_LENGTH)

```

## Déclaration Visual Basic pour MQCIH

```

Type MQCIH
  StrucId           As String*4 'Structure identifier'
  Version           As Long     'Structure version number'
  StrucLength       As Long     'Length of MQCIH structure'
  Encoding          As Long     'Reserved'
  CodedCharSetId   As Long     'Reserved'
  Format            As String*8  'MQ format name of data that follows'
  'MQCIH'
  Flags            As Long     'Flags'
  ReturnCode       As Long     'Return code from bridge'
  CompCode         As Long     'MQ completion code or CICS EIBRESP'
  Reason           As Long     'MQ reason or feedback code, or CICS'
  'EIBRESP2'
  UOWControl       As Long     'Unit-of-work control'
  GetWaitInterval  As Long     'Wait interval for MQGET call issued'
  'by bridge task'
  LinkType         As Long     'Link type'
  OutputDataLength As Long     'Output COMMAREA data length'
  FacilityKeepTime As Long     'Bridge facility release time'
  ADSDescriptor    As Long     'Send/receive ADS descriptor'
  ConversationalTask As Long   'Whether task can be conversational'
  TaskEndStatus    As Long     'Status at end of task'
  Facility         As MQBYTE8  'Bridge facility token'
  Function         As String*4  'MQ call name or CICS EIBFN function'
  AbendCode        As String*4  'Abend code'
  Authenticator    As String*8  'Password or passticket'
  Reserved1        As String*8  'Reserved'

```

ReplyToFormat	As String*8	'MQ format name of reply message'
RemoteSysId	As String*4	'Reserved'
RemoteTransId	As String*4	'Reserved'
TransactionId	As String*4	'Transaction to attach'
FacilityLike	As String*4	'Terminal emulated attributes'
AttentionId	As String*4	'AID key'
StartCode	As String*4	'Transaction start code'
CancelCode	As String*4	'Abend transaction code'
NextTransactionId	As String*4	'Next transaction to attach'
Reserved2	As String*8	'Reserved'
Reserved3	As String*8	'Reserved'
CursorPosition	As Long	'Cursor position'
ErrorOffset	As Long	'Offset of error in message'
InputItem	As Long	'Reserved'
Reserved4	As Long	'Reserved'
End Type		

### ***StrucId (MQCHAR4) pour MQCIH***

Il s'agit de l'identificateur de structure de la structure d'en-tête d'informations CICS . Il s'agit toujours d'une zone d'entrée. Sa valeur est MQCIH\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUC\_MQCIH**

Identificateur de la structure d'en-tête d'informations CICS .

Pour le langage de programmation C, la constante MQCIH\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQCIH\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### ***Version (MQLONG) pour MQCIH***

Cette zone est une zone de demande. Sa valeur initiale est MQCIH\_VERSION\_2.

Il doit s'agir de l'une des valeurs suivantes :

#### **MQCIH\_VERSION\_1**

Structure d'en-tête d'information Version-1 CICS .

#### **MQCIH\_VERSION\_2**

Structure d'en-tête d'information Version-2 CICS .

Les zones qui existent uniquement dans la version la plus récente de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

#### **VERSION\_MQCIH\_CURRENT\_VERSION**

Version actuelle de la structure d'en-tête d'informations CICS .

### ***StrucLength (MQLONG) pour MQCIH***

Cette zone est une zone de demande, avec la valeur initiale MQCIH\_LENGTH\_2.

Il doit s'agir de l'une des valeurs suivantes :

#### **MQCIH\_LENGTH\_1**

Longueur de la structure d'en-tête d'information version-1 CICS .

#### **MQCIH\_LENGTH\_2**

Longueur de la structure d'en-tête d'information version-2 CICS .

La constante suivante indique la longueur de la version en cours:

#### **MQCIH\_LONGUEUR\_EN\_COURS**

Longueur de la version actuelle de la structure d'en-tête d'informations CICS .

### ***Codage (MQLONG) pour MQCIH***

Cette zone est une zone réservée ; sa valeur n'est pas significative. Sa valeur initiale est 0.

Le codage des structures prises en charge qui suivent une structure MQCIH est identique au codage de la structure MQCIH elle-même et provient de tout en-tête IBM MQ précédent.

### **CodedCharSetId (MQLONG) pour MQCIH**

CodedCharSetId est une zone réservée ; sa valeur n'est pas significative. La valeur initiale de cette zone est 0.

L'ID de jeu de caractères pour les structures prises en charge qui suivent une structure MQCIH est identique à l'ID de jeu de caractères de la structure MQCIH elle-même et provient de tout en-tête IBM MQ précédent.

### **Format (MQCHAR8) pour MQCIH**

Cette zone affiche le nom de format IBM MQ des données qui suivent la structure MQCIH.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de codage de la zone *Format* dans MQMD.

Ce nom de format est également utilisé pour le message de réponse si la zone *ReplyToFormat* a la valeur MQFMT\_NONE.

- Pour les demandes DPL, *Format* doit être le nom de format de la zone de communication.
- Pour les demandes 3270, *Format* doit être CSQCBDCI et le pont définit le format sur CSQCBDCO pour les messages de réponse.

Les exits de conversion de données pour ces formats doivent être installés sur le gestionnaire de files d'attente où ils doivent être exécutés.

Si le message de demande génère un message de réponse d'erreur, le nom de format du message de réponse d'erreur est MQFMT\_STRING.

Cette zone est une zone de demande. La longueur de cette zone est indiquée par MQ\_FORMAT\_LENGTH. La valeur initiale de cette zone est MQFMT\_NONE.

### **Indicateurs (MQLONG) pour MQCIH**

Cette zone est une zone de demande. La valeur initiale de cette zone est MQCIH\_NONE.

La valeur doit être:

#### **MQCIH\_NONE**

Aucun indicateur.

#### **EXPIRATION\_MOT\_DE\_PASSE\_MQCIH**

Le message de réponse contient:

- Les mêmes options de rapport d'expiration que le message de demande.
- Heure d'expiration restante du message de demande sans ajustement effectué pour le temps de traitement du pont.

Si vous omettez cette valeur, le délai d'expiration est défini sur *unlimited*.

#### **MQCIH\_REPLY\_WITHOUT\_NULLS**

La longueur du message de réponse d'une demande de programme DPL CICS est ajustée pour exclure les valeurs nulles de fin (X'00') à la fin de la zone de communication renvoyée par le programme DPL. Si cette valeur n'est pas définie, les valeurs nulles peuvent être significatives et la zone de communication complète est renvoyée.

#### **MQCIH\_SYNC\_ON\_RETURN**

La liaison CICS pour les demandes DPL utilise l'option SYNCONRETURN, ce qui amène CICS à prendre un point de synchronisation lorsque le programme se termine s'il est envoyé à une autre région CICS . Le pont ne spécifie pas la région CICS à laquelle la demande doit être envoyée ; elle est contrôlée par les fonctions d'équilibrage de charge ou de définition de programme CICS .

### **ReturnCode (MQLONG) pour MQCIH**

La valeur de cette zone est le code retour du CICS bridge décrivant le résultat du traitement effectué par le pont. Cette zone est une zone de réponse, avec une valeur initiale de MQCRC\_OK.

Les zones *Function*, *CompCode*, *Reasonet AbendCode* peuvent contenir des informations supplémentaires (voir [Tableau 476](#), à la page 308). La valeur est l'une des suivantes :

**MQCRC\_APPLICATION\_ABEND**

(5, X'005') L'application s'est terminée de manière anormale.

**MQCRC\_BRIDGE\_ABEND**

(4, X'004') CICS bridge s'est arrêté de manière anormale.

**MQCRC\_BRIDGE\_ERREUR**

(3, X'003') CICS bridge a détecté une erreur.

**MQCRC\_DÉLAI\_BRIDGE\_ATTENTE**

(8, X'008') Deuxième message ou message ultérieur dans l'unité d'oeuvre en cours qui n'a pas été reçu dans le délai imparti.

**MQCRC\_CICS\_EXEC\_ERROR**

(1, X'001') L'instruction EXEC CICS a détecté une erreur.

**MQCRC\_MQ\_API\_ERREUR**

(2, X'002') L'appel MQ a détecté une erreur.

**MQCRC\_OK**

(0, X'000') Aucune erreur.

**MQCRC\_PROGRAM\_NOT\_AVAILABLE**

(7, X'007') Programme non disponible.

**MQCRC\_SECURITY\_ERROR**

(6, X'006') Une erreur de sécurité s'est produite.

**MQCRC\_TRANSID\_NON DISPONIBLE**

(9, X'009') Transaction non disponible.

***CompCode (MQLONG) pour MQCIH***

Cette zone est une zone de réponse. Sa valeur initiale est MQCC\_OK

La valeur renvoyée dans cette zone dépend de *ReturnCode* ; voir [Tableau 476](#), à la page 308.

***Motif (MQLONG) pour MQCIH***

Cette zone est une zone de réponse. Sa valeur initiale est MQRC\_NONE.

La valeur renvoyée dans cette zone dépend de *ReturnCode* ; voir [Tableau 476](#), à la page 308.

***UOWControl (MQLONG) pour MQCIH***

Cette zone est une zone de demande qui contrôle le traitement de l'unité de travail effectué par CICS bridge. La valeur initiale de cette zone est MQCUOWC\_ONLY.

Vous pouvez demander au pont d'exécuter une seule transaction ou un ou plusieurs programmes au sein d'une unité de travail. La zone indique si CICS bridge démarre une unité de travail, exécute la fonction demandée dans l'unité de travail en cours ou met fin à l'unité de travail en la validant ou en la sauvegardant. Diverses combinaisons sont prises en charge, pour optimiser les flux de transmission de données.

Il doit s'agir de l'une des valeurs suivantes :

**MQCUOWC\_ONLY**

Démarrez l'unité de travail, exécutez la fonction, puis validez l'unité de travail.

**MQCUOWC\_CONTINUE**

Données supplémentaires pour l'unité d'oeuvre en cours (3270 uniquement).

**MQCUOWC\_FIRST**

Démarrer l'unité de travail et exécuter la fonction.

**MQCUOWC\_MILIEU**

Exécuter la fonction dans l'unité de travail en cours

### **MQCUOWC\_LAST**

Exécutez la fonction, puis validez l'unité de travail.

### **MQCUOWC\_COMMIT**

Validez l'unité d'oeuvre (DPL uniquement).

### **MQCUOWC\_BACKOUT**

Revenez à l'unité d'oeuvre (DPL uniquement).

### ***Intervalle GetWait(MQLONG) pour MQCIH***

Cette zone est une zone de demande. Sa valeur initiale est MQCGWI\_DEFAULT.

Cette zone s'applique uniquement lorsque *UOWControl* a la valeur MQCUOWC\_FIRST. Elle permet à l'application émettrice de spécifier la durée approximative, en millisecondes, pendant laquelle les appels MQGET émis par le pont attendent les deuxième et suivants messages de demande pour l'unité de travail démarrée par ce message. Cette fonction remplace l'intervalle d'attente par défaut utilisé par le pont.

Vous pouvez utiliser les valeurs spéciales suivantes:

### **MQCGWI\_VALEUR PAR DEFAUT**

Intervalle d'attente par défaut.

Cette valeur permet à CICS bridge d'attendre le temps spécifié lors du démarrage du pont.

### **MQWI\_ILLIMITÉ**

Intervalle d'attente illimité.

### ***LinkType (MQLONG) pour MQCIH***

Cette zone est une zone de demande. Sa valeur initiale est MQCLT\_PROGRAM.

Cette valeur indique le type d'objet que le pont tente de lier. Il doit s'agir de l'une des valeurs suivantes:

### **PROGRAMME MQCL**

Programme DPL.

### **TRANSACTION MQCL**

Transaction 3270.

### ***OutputDataLongueur (MQLONG) pour MQCIH***

Cette zone est une zone de demande utilisée uniquement pour les programmes DPL. Sa valeur initiale est MQCODL\_AS\_INPUT.

Cette valeur correspond à la longueur des données utilisateur à renvoyer au client dans un message de réponse. Cette longueur inclut le nom de programme à 8 octets. La longueur de la zone de communication transmise au programme lié est la longueur maximale de cette zone et la longueur des données utilisateur du message de demande, moins 8.

**Remarque :** La longueur des données utilisateur dans un message correspond à la longueur du message, à l'exclusion de la structure MQCIH.

Si la longueur des données utilisateur dans le message de demande est inférieure à *OutputDataLength*, l'option DATALENGTH de la commande LINK est utilisée, ce qui permet au LINK d'être expédié efficacement à une autre région CICS .

Vous pouvez utiliser la valeur spéciale suivante:

### **MQCODL\_AS\_ENTREE**

La longueur de sortie est identique à la longueur d'entrée.

Cette valeur peut être nécessaire même si aucune réponse n'est demandée, afin de s'assurer que la zone de communication transmise au programme lié est de taille suffisante.

### ***FacilityKeepTime (MQLONG) pour MQCIH***

FacilityKeep: durée, en secondes, pendant laquelle la fonction de pont est conservée après la fin de la transaction utilisateur.

Pour les transactions pseudo-conversationnelles, indiquez une valeur qui correspond à la durée prévue d'une pseudo-conversation ; indiquez zéro pour la dernière transaction d'une pseudo-conversation et zéro pour les autres types de transaction.

Cette zone est une zone de demande utilisée uniquement pour les transactions 3270. La valeur initiale de cette zone est 0.

### ***ADSDescriptor (MQLONG) pour MQCIH***

Cette zone indique si les descripteurs ADS doivent être envoyés sur les demandes SEND et RECEIVE BMS.

Les valeurs suivantes sont définies :

#### **MQCADSD\_NONE**

N'envoyez pas ou ne recevez pas de descripteurs ADS.

#### **MQCADSD\_SEND**

Envoi de descripteurs ADS.

#### **MQCADSD\_RECV**

Descripteurs ADS de réception.

#### **MQCADSD\_MSGFORMAT**

Utilisez le format de message pour les descripteurs ADS.

Ceci permet d'envoyer ou de recevoir les descripteurs ADS en utilisant la forme longue du descripteur ADS. Le format long comporte des zones alignées sur des limites de 4 octets.

Définissez la zone *ADSDescriptor* comme suit:

- Si vous n'utilisez pas de descripteurs ADS, définissez la zone sur MQCADSD\_NONE.
- Si vous utilisez des descripteurs ADS avec le *même* CCSID dans chaque environnement, définissez la zone sur la somme de MQCADSD\_SEND et MQCADSD\_RECV.
- Si vous utilisez des descripteurs ADS avec des CCSID *différents* dans chaque environnement, définissez la zone sur la somme de MQCADSD\_SEND, MQCADSD\_RECV et MQCADSD\_MSGFORMAT.

Il s'agit d'une zone de demande utilisée uniquement pour les transactions 3270. La valeur initiale de cette zone est MQCADSD\_NONE.

### ***ConversationalTask (MQLONG) pour MQCIH***

Cette zone indique si la tâche doit être autorisée à émettre des demandes d'informations supplémentaires ou à arrêter la tâche et à émettre un message de fin anormale.

La valeur doit être l'une des options suivantes:

#### **MQCCT\_OUI**

La tâche est conversationnelle.

#### **MQCCT\_NO**

La tâche n'est pas conversationnelle.

Cette zone est une zone de demande utilisée uniquement pour les transactions 3270. La valeur initiale de cette zone est MQCCT\_NO.

### ***Statut TaskEnd(MQLONG) pour MQCIH***

Cette zone est une zone de réponse indiquant le statut de la transaction utilisateur à la fin de la tâche. La zone est utilisée uniquement pour les transactions 3270 et sa valeur initiale est MQCTES\_NOSYNC.

L'une des valeurs suivantes est renvoyée:

#### **MQCTES\_NOSYNC**

Non synchronisé.

La transaction utilisateur n'est pas encore terminée et n'est pas synchronisée. La zone *MsgType* dans MQMD est MQMT\_REQUEST dans ce cas.

#### **MQCTES\_COMMIT**

Validation de l'unité de travail.

La transaction utilisateur n'est pas encore terminée, mais elle a synchronisé la première unité de travail. La zone *MsgType* dans MQMD est MQMT\_DATAGRAM dans ce cas.

#### **MQCTES\_BACKOUT**

Unité de travail d'arrière-système.

La transaction utilisateur n'est pas encore terminée. L'unité de travail en cours est annulée. La zone *MsgType* dans MQMD est MQMT\_DATAGRAM dans ce cas.

#### **MQCTES\_ENDTASK**

Fin de la tâche.

La transaction utilisateur s'est arrêtée (ou s'est arrêtée de manière anormale). La zone *MsgType* dans MQMD est MQMT\_REPLY dans ce cas.

### **Fonction (MQBYTE8) pour MQCIH**

Cette zone affiche le jeton de la fonction de pont de 8 octets.

Un jeton de fonction de pont permet à plusieurs transactions d'une pseudo-conversation d'utiliser la même fonction de pont (terminal 3270 virtuel). Dans le premier ou le seul message d'une pseudo-conversation, définissez la valeur MQCFAC\_NONE. Cette valeur indique à CICS d'allouer une nouvelle fonction de pont pour ce message. Un jeton de fonction de pont est renvoyé dans les messages de réponse lorsqu'un *FacilityKeepTime* différent de zéro est spécifié dans le message d'entrée. Les messages d'entrée suivants dans une pseudo-conversation doivent ensuite utiliser le même jeton de fonction de pont.

La valeur spéciale suivante est définie:

#### **MQCFAC\_NONE**

Aucun jeton de fonction spécifié.

Pour le langage de programmation C, la constante MQCFAC\_NONE\_ARRAY est également définie et a la même valeur que MQCFAC\_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Cette zone est à la fois une zone de demande et une zone de réponse utilisées uniquement pour les transactions 3270. La longueur de cette zone est indiquée par MQ\_FACILITY\_LENGTH. La valeur initiale de cette zone est MQCFAC\_NONE.

### **Fonction (MQCHAR4) pour MQCIH**

Cette zone est une zone de réponse. La longueur de cette zone est indiquée par MQ\_FUNCTION\_LENGTH. La valeur initiale de cette zone est MQCFUNC\_NONE.

La valeur renvoyée dans cette zone dépend de *ReturnCode* ; voir [Tableau 476](#), à la [page 308](#). Les valeurs suivantes sont possibles lorsque *Function* contient un nom d'appel IBM MQ :

#### **MQCFUNC\_MQCONN**

Appel MQCONN.

#### **MQCFUNC\_MQGET**

Appel MQGET.

#### **MQCFUNC\_MQINQ**

Appel MQINQ.

#### **MQCFUNC\_MQOPEN**

Appel MQOPEN.

#### **MQCFUNC\_MQPUT**

Appel MQPUT.

#### **MQCFUNC\_MQPUT1**

Appel MQPUT1 .

#### **MQCFUNC\_NONE**

Pas d'appel.

Dans tous les cas, pour le langage de programmation C, les constantes MQCFUNC\_\*\_ARRAY sont également définies ; ces constantes ont les mêmes valeurs que les constantes MQCFUNC\_\* correspondantes, mais sont des tableaux de caractères au lieu de chaînes.

### **AbendCode (MQCHAR4) pour MQCIH**

AbendCode est une zone de réponse. La longueur de cette zone est indiquée par MQ\_ABEND\_CODE\_LENGTH. La valeur initiale de cette zone est de 4 caractères blancs.

La valeur renvoyée dans cette zone est significative uniquement si la zone *ReturnCode* a la valeur MQCRC\_APPLICATION\_ABEND ou MQCRC\_BRIDGE\_ABEND. Si tel est le cas, *AbendCode* contient la valeur ABCODE CICS .

### **Authentificateur (MQCHAR8) pour MQCIH**

La valeur de cette zone est le mot de passe ou le mot de passe.

Si l'authentification par identificateur d'utilisateur est active pour CICS bridge, *Authenticator* est utilisé avec l'identificateur d'utilisateur dans le contexte d'identité MQMD pour authentifier l'expéditeur du message.

Il s'agit d'une zone de demande. La longueur de cette zone est indiquée par MQ\_AUTHENTICATOR\_LENGTH. La valeur initiale de cette zone est 8 blancs.

### **Reserved1 (MQCHAR8) pour MQCIH**

Cette zone est une zone réservée. La valeur doit être à 8 blancs.

### **Format ReplyTo(MQCHAR8) pour MQCIH**

La valeur de cette zone est le nom de format IBM MQ du message de réponse envoyé en réponse au message en cours.

Les règles de codage de cette zone sont les mêmes que celles de codage de la zone *Format* dans MQMD.

Cette zone est une zone de demande utilisée uniquement pour les programmes DPL. La longueur de cette zone est indiquée par MQ\_FORMAT\_LENGTH. La valeur initiale de cette zone est MQFMT\_NONE.

### **ID RemoteSys(MQCHAR4) pour MQCIH**

Cette zone affiche l'identificateur système CICS du système CICS qui traite la demande.

Si cette zone est vide, la demande système CICS est traitée sur le même système CICS que le moniteur de pont. Le SYSID utilisé est renvoyé dans le message de réponse.

Pour une pseudo-conversation 3270, tous les messages suivants de la conversation doivent spécifier le SYSID distant renvoyé dans la réponse initiale. S'il est spécifié, le SYSID doit:

- Soyez actif.
- Accédez à la file d'attente des demandes IBM MQ .
- Soyez accessible par les liaisons CICS ISC à partir du système CICS du moniteur de pont.

### **ID RemoteTrans(MQCHAR4) pour MQCIH**

Cette zone est facultative. La longueur de cette zone est indiquée par MQ\_TRANSACTION\_ID\_LENGTH.

Si elle est spécifiée, la zone est utilisée comme valeur RTRANSID de CICS START.

### **TransactionId (MQCHAR4) pour MQCIH**

Cette zone est une zone de demande. Sa longueur est donnée par MQ\_TRANSACTION\_ID\_LENGTH. La valeur initiale de cette zone est de quatre blancs.

Si *LinkType* a la valeur MQCLT\_TRANSACTION, *TransactionId* est l'identificateur de transaction de la transaction utilisateur à exécuter ; indiquez une valeur non vide dans ce cas.

Si *LinkType* a la valeur MQCLT\_PROGRAM, *TransactionId* est le code de transaction sous lequel tous les programmes de l'unité de travail doivent être exécutés. Si vous indiquez une valeur vide, le code



de transaction par défaut du pont DPL CICS (CKBP) est utilisé. Si la valeur n'est pas vide, vous devez l'avoir définie sur CICS en tant que transaction locale avec un programme initial CSQCBP00. Cette zone s'applique uniquement lorsque *UOWControl* a la valeur MQCUOWC\_FIRST ou MQCUOWC\_ONLY.

### **FacilityLike (MQCHAR4) pour MQCIH**

FacilityLike est le nom d'un terminal installé qui doit être utilisé comme modèle pour la fonction de pont.

Une valeur vide signifie que *FacilityLike* est extraite de la définition de profil de transaction de pont ou qu'une valeur par défaut est utilisée.

Cette zone est une zone de demande utilisée uniquement pour les transactions 3270. La longueur de cette zone est indiquée par MQ\_FACILITY\_LIKE\_LENGTH. La valeur initiale de cette zone est de quatre blancs.

### **AttentionId (MQCHAR4) pour MQCIH**

La valeur de cette zone détermine la valeur initiale de la clé AID au démarrage de la transaction. Il s'agit d'une valeur de 1 octet, alignée à gauche.

AttentionId est une zone de demande utilisée uniquement pour les transactions 3270. La longueur de cette zone est indiquée par MQ\_ATTENTION\_ID\_LENGTH. La valeur initiale de cette zone est de quatre blancs.

### **StartCode (MQCHAR4) pour MQCIH**

La valeur de cette zone est un indicateur spécifiant si le pont émule une transaction de terminal ou une transaction lancée avec START.

Il doit s'agir de l'une des valeurs suivantes :

#### **MQCSC\_DEBUT**

Début.

#### **Données MQCSC\_STARTDATA**

Données de démarrage.

#### **PUT MQCSC\_TERMINE**

Entrée de terminal.

#### **MQCSC\_NONE**

Néant.

Dans tous les cas, pour le langage de programmation C, les constantes MQCSC\_\*\_ARRAY sont également définies ; ces constantes ont les mêmes valeurs que les constantes MQCSC\_\* correspondantes, mais sont des tableaux de caractères au lieu de chaînes.

Dans la réponse du pont, cette zone est définie sur le code de début correspondant à l'ID transaction suivant contenu dans la zone *NextTransactionId*. Les codes de début suivants sont possibles dans la réponse:

- MQCSC\_DEBUT
- Données MQCSC\_STARTDATA
- PUT MQCSC\_TERMINE

Pour CICS Transaction Server 1.2, cette zone est une zone de demande uniquement ; sa valeur dans la réponse n'est pas définie.

Pour CICS Transaction Server 1.3 et les éditions suivantes, cette zone est à la fois une zone de demande et une zone de réponse.

Cette zone est utilisée uniquement pour les transactions 3270. La longueur de cette zone est indiquée par MQ\_START\_CODE\_LENGTH. La valeur initiale de cette zone est MQCSC\_NONE.

### **CancelCode (MQCHAR4) pour MQCIH**

La valeur de cette zone est le code de fin anormale à utiliser pour mettre fin à la transaction (normalement une transaction conversationnelle qui demande plus de données). Sinon, cette zone est mise à blanc.

Cette zone est une zone de demande utilisée uniquement pour les transactions 3270. La longueur de cette zone est indiquée par MQ\_CANCEL\_CODE\_LENGTH. La valeur initiale de cette zone est de quatre blancs.

### **ID NextTransaction(MQCHAR4) pour MQCIH**

Cette valeur correspond au nom de la transaction suivante renvoyée par la transaction utilisateur (généralement par EXEC CICS RETURN TRANSID). S'il n'y a pas de transaction suivante, cette zone est mise à blanc.

Cette zone est une zone de réponse utilisée uniquement pour les transactions 3270. La longueur de cette zone est indiquée par MQ\_TRANSACTION\_ID\_LENGTH. La valeur initiale de cette zone est de quatre blancs.

### **Reserved2 (MQCHAR8) pour MQCIH**

Cette zone est une zone réservée. La valeur doit être à 8 blancs.

### **Reserved3 (MQCHAR8) pour MQCIH**

Cette zone est une zone réservée. La valeur doit être à 8 blancs.

### **CursorPosition (MQLONG) pour MQCIH**

La valeur de cette zone indique la position initiale du curseur lorsque la transaction est démarrée. Pour les transactions conversationnelles, la position du curseur est dans le vecteur RECEIVE.

Cette zone est une zone de demande utilisée uniquement pour les transactions 3270. La valeur initiale de cette zone est 0. Cette zone n'est pas présente si *Version* est inférieur à MQCIH\_VERSION\_2.

### **ErrorOffset (MQLONG) pour MQCIH**

La zone ErrorOffset affiche la position des données non valides détectées par l'exit de pont. Cette zone indique le décalage entre le début du message et l'emplacement des données non valides.

ErrorOffset est une zone de réponse utilisée uniquement pour les transactions 3270. La valeur initiale de cette zone est 0. Cette zone n'est pas présente si *Version* est inférieur à MQCIH\_VERSION\_2.

### **InputItem (MQLONG) pour MQCIH**

Cette zone est une zone réservée. La valeur doit être 0.

Cette zone n'est pas présente si *Version* est inférieur à MQCIH\_VERSION\_2.

### **Reserved4 (MQLONG) pour MQCIH**

Cette zone est une zone réservée. La valeur doit être 0.

Cette zone n'est pas présente si *Version* est inférieur à MQCIH\_VERSION\_2.

## **MQCMHO-Options de création de descripteur de message**

La structure **MQCMHO** permet aux applications de spécifier des options qui contrôlent la façon dont les descripteurs de message sont créés. La structure est un paramètre d'entrée dans l'appel **MQCRTMH**.

## Disponibilité

La structure **MQCMHO** est disponible sur les plateformes suivantes:

-  AIX
-  IBM i
-  Linux
-  Windows
-  z/OS

et pour IBM MQ MQI clients connecté à ces systèmes.

## Jeu de caractères et codage

Les données dans **MQCMHO** doivent être dans le jeu de caractères de l'application et dans le codage de l'application ( **MQENC\_NATIVE** ).

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID_STRUC_MQCMHO_	'CMHO'
<u>Version</u> (numéro de version de la structure)	MQCMHO_VERSION_1	1
<u>Options</u> (options)	MQCMHO_VALEUR_PAR _DÉFAUT_VALIDATION	0

**Remarques :**

1. Dans le langage de programmation C, la variable macro **MQCMHO\_DEFAULT** contient les valeurs répertoriées dans la table. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQCMHO MyCMHO = {MQCMHO_DEFAULT};
```

## Déclarations de langue

Déclaration C pour MQCMHO

```
struct tagMQCMHO {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    Options;          /* Options that control the action of MQCRTMH */
};
```

Déclaration COBOL pour MQCMHO

```
** MQCMHO structure
   10 MQCMHO.
** Structure identifier
```

```

15 MQCMHO-STRUCID      PIC X(4).
** Structure version number
15 MQCMHO-VERSION      PIC S9(9) BINARY.
** Options that control the action of MQCRTMH
15 MQCMHO-OPTIONS      PIC S9(9) BINARY.

```

### Déclaration PL/I pour MQCMHO

```

dcl
  1 MQCMHO based,
  3 StrucId      char(4),          /* Structure identifier */
  3 Version      fixed bin(31), /* Structure version number */
  3 Options      fixed bin(31), /* Options that control the action of MQCRTMH */

```

### Déclaration High Level Assembler pour MQCMHO

```

MQCMHO          DSECT
MQCMHO_STRUCID  DS   CL4   Structure identifier
MQCMHO_VERSION  DS   F     Structure version number
MQCMHO_OPTIONS  DS   F     Options that control the action of
*                MQCRTMH
MQCMHO_LENGTH   EQU   *-MQCMHO
MQCMHO_AREA     DS   CL(MQCMHO_LENGTH)

```

### **StrucId (MQCHAR4) pour MQCMHO**

Il s'agit de l'identificateur de structure de la structure d'options de création de descripteur de message. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQCMHO\_STRUC\_ID.

La valeur doit être:

#### **MQCMHO\_STRUC\_ID**

Identificateur de la structure des options de création de descripteur de message.

Pour le langage de programmation C, la constante **MQCMHO\_STRUC\_ID\_ARRAY** est également définie. Il a la même valeur que **MQCMHO\_STRUC\_ID**, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQCMHO**

Cette zone est toujours une zone d'entrée. Sa valeur initiale est MQCMHO\_VERSION\_1.

Il s'agit du numéro de version de la structure ; la valeur doit être:

#### **MQCMHO\_VERSION\_1**

Version-1 crée une structure d'options de descripteur de message.

La constante suivante indique le numéro de version de la version en cours:

#### **MQCMHO\_CURRENT\_VERSION**

Version actuelle de la structure des options de création de descripteur de message.

### **Options (MQLONG) pour MQCMHO**

Cette zone est toujours une zone d'entrée. Sa valeur initiale est MQCMHO\_DEFAULT\_VALIDATION.

L'une des options suivantes peut être spécifiée:

#### **MQCMHO\_VALIDATION**

Lorsque **MQSETMP** est appelé pour définir une propriété dans ce descripteur de message, le nom de la propriété est validé pour s'assurer qu'il:

- ne contient pas de caractères non valides.

- ne commence pas par JMS ou usr.JMS sauf pour les éléments suivants:
  - JMSCorrelationID
  - JMSReplyTo
  - JMSType
  - JMSXGroupID
  - JMSXGroupSeq

Ces noms sont réservés aux propriétés JMS .

- n'est pas l'un des mots clés suivants, en majuscules ou en minuscules:
  - ET
  - ENTRE
  - ESCAPE
  - FALSE
  - IN
  - support IBM
  - SIMILAIRE A
  - N'EST PAS
  - NULL
  - OU
  - exit utilisateur associé à une tâche
- ne commence pas le corps. ou root. (sauf pour Root.MQMD.)

Si la propriété est MQ-defined (mq. \*) et que le nom est reconnu, les zones de descripteur de propriété sont définies sur les valeurs correctes pour la propriété. Si la propriété n'est pas reconnue, la zone *Support* du descripteur de propriété est définie sur **MQPD\_OPTIONAL**.

### **MQCMHO\_VALEUR\_PAR\_DÉFAUT**

Cette valeur indique que le niveau de validation par défaut des noms de propriété est atteint.

Le niveau de validation par défaut est équivalent au niveau spécifié par **MQCMHO\_VALIDATE**.

Cette valeur est la valeur par défaut.

### **MQCMHO\_NO\_VALIDATION**

Aucune validation n'est effectuée sur le nom de la propriété. Voir la description de **MQCMHO\_VALIDATE**.

**Option par défaut:** Si aucune des options décrites ci-dessus n'est requise, l'option suivante peut être utilisée:

### **MQCMHO\_AUCUN**

Toutes les options prennent leurs valeurs par défaut. Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. **MQCMHO\_NONE** aide la documentation du programme ; il n'est pas prévu que cette option soit utilisée avec d'autres, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

## **MQCNO-Options de connexion**

La structure MQCNO permet à l'application de spécifier des options relatives à la connexion au gestionnaire de files d'attente. La structure est un paramètre d'entrée-sortie sur l'appel MQCONN.

Pour plus d'informations sur l'utilisation des descripteurs partagés et de l'appel MQCONN, voir Connexions partagées (indépendantes de l'unité d'exécution) avec MQCONN.

## Disponibilité

La structure MQCNO est disponible sur les plateformes suivantes:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour IBM MQ MQI clients connecté à ces systèmes.

## Version

Les fichiers d'en-tête, COPY et INCLUDE fournis pour les langages de programmation pris en charge contiennent la version la plus récente de MQCNO, mais avec la valeur initiale de la zone *Version* définie sur MQCNO\_VERSION\_1. Pour utiliser des zones qui ne sont pas présentes dans la structure version-1, l'application doit définir la zone *Version* sur le numéro de version requis.

## Jeu de caractères et codage

Les données de MQCNO doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et le codage du gestionnaire de files d'attente local indiqué par MQENC\_NATIVE. Toutefois, si l'application s'exécute en tant que IBM MQ MQI client, la structure doit être dans le jeu de caractères et le codage du client.

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Tableau 479. Zones de MQCNO pour MQCNO		
Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID_STRUC_MQCNO	'CNO~'
<u>Version</u> (numéro de version de la structure)	MQCNO_VERSION_1	1
<u>Options</u> (options qui contrôlent l'action de MQCONNX)	MQCNO_AUCUN	0
<b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieur à MQCNO_VERSION_2.		
<u>ClientConnDécalage</u> (décalage de la structure MQCD pour la connexion client)	Aucun	0
<u>ClientConnPtr</u> (adresse de la structure MQCD pour la connexion client)	Aucun	Pointeur null ou octets null
<b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieur à MQCNO_VERSION_3.		
<u>ConnTag</u> (balise de connexion du gestionnaire de files d'attente)	MQCT_AUCUN	Valeurs NULL
<b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieur à MQCNO_VERSION_4.		

Tableau 479. Zones de MQCNO pour MQCNO (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>SSLConfigPtr</u> (adresse de la structure MQSCO pour la connexion client)	Aucun	Pointeur null ou octets null
<u>SSLConfigOffset</u> (décalage de la structure MQSCO pour la connexion client)	Aucun	0
<b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieur à MQCNO_VERSION_5.		
<u>ConnectionId</u> (ID de connexion unique)	Aucun	Pointeur null ou octets null
<u>SecurityParmsDécalage</u> (décalage de la structure MQSCO pour les paramètres de sécurité)	Aucun	Pointeur null ou octets null
<u>SecurityParmsPtr</u> (adresse de la structure MQSCO pour les paramètres de sécurité)	Aucun	Pointeur null ou octets null
<b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieur à MQCNO_VERSION_6.		
<u>Réservé</u> (zone réservée)	Aucun	Zone réservée pour remplir la structure jusqu'à une limite de 64 bits.
<u>CCDTUrlLength</u> (longueur de l' URL de la table de définition de canal du client)	Aucun	Longueur de la chaîne identifiée par <i>CCDTUrlPtr</i> ou <i>CCDTUrlOffset</i>
<u>CCDTUrlPtr</u> (pointeur CCDT URL )	Aucun	Pointeur vers une chaîne contenant une URL, pour identifier l'emplacement de la table de canaux de connexion client à utiliser pour la connexion.
<u>CCDTUrlOffset</u> (décalage URL CCDT)	Aucun	Décalage en octets à partir d'une chaîne contenant une URL qui identifie l'emplacement de la table de canaux de connexion client à utiliser pour la connexion.
<b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieur à MQCNO_VERSION_7.		
<u>ApplName</u> (nom défini par l'application)	Aucun	Nom défini par l'application pour identifier la connexion au gestionnaire de files d'attente

Tableau 479. Zones de MQCNO pour MQCNO (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>Reserved2</u> (zone réservée)	Aucun	Zone réservée pour remplir la structure jusqu'à une limite de 64 bits.
<div style="display: flex; justify-content: space-between; align-items: center;"> <span style="background-color: #0056b3; color: white; padding: 2px 5px;">&gt; V9.3.0</span> <span style="background-color: #0056b3; color: white; padding: 2px 5px;">&gt; V9.3.0</span> </div> <p><b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieur à MQCNO_VERSION_8.</p>		
<span style="background-color: #0056b3; color: white; padding: 2px 5px;">&gt; V9.3.0</span> <u>BalanceParms</u>   Décalage	Aucun	Décalage en octets de la structure MQBNO
<span style="background-color: #0056b3; color: white; padding: 2px 5px;">&gt; V9.3.0</span> <u>BalanceParmsPtr</u>	Aucun	Pointeur vers l'emplacement de la structure MQBNO
<p><b>Remarques :</b></p> <ol style="list-style-type: none"> <li>Le symbole ~ représente un caractère blanc unique.</li> <li>Dans le langage de programmation C, la variable macro MQCNO_DEFAULT contient les valeurs répertoriées dans la table. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</li> </ol> <div style="background-color: #f0f0f0; padding: 10px; margin-top: 10px;"> <pre>MQCNO MycNO = {MQCNO_DEFAULT};</pre> </div>		

## Déclarations de langue

**Remarque :** > V9.3.0 Dans chacune des déclarations suivantes, les deux dernières lignes (Offset of the MQBMO structure et Address of the location of the MQBMO structure) ont été ajoutées à IBM MQ 9.2.4 pour les utilisateurs de CD et à IBM MQ 9.3.0 pour les utilisateurs de LTS.

Déclaration C pour MQCNO

```
typedef struct tagMQCNO MQCNO;
struct tagMQCNO {
    MQCHAR4    StructId;           /* Structure identifier */
    MQLONG     Version;           /* Structure version number */
    MQLONG     Options;           /* Options that control the action of
    MQCONNX */
    MQLONG     ClientConnOffset; /* Offset of MQCD structure for client
    connection */
    MQPTR      ClientConnPtr;     /* Address of MQCD structure for client
    connection */
    MQBYTE128  ConnTag;           /* Queue manager connection tag */
    PMQSCO     SSLConfigPtr;     /* Address of MQSCO structure for client
    connection */
    MQLONG     SSLConfigOffset; /* Offset of MQSCO structure for client
    connection */
    MQBYTE24   ConnectionId;     /* Unique connection identifier */
    MQLONG     SecurityParmsOffset /* Security fields */
    PMQCSP     SecurityParmsPtr /* Security parameters */
    MQLONG     CCDTUrlLength     /* Length of string identified by Ptr or offset */
    MQLONG     CCDTUrlOffset     /* Offset in bytes to URL of client connection channel */
    PMQURL     CCDTUrlPtr       /* Address of string containing URL */
    MQBYTE4    Reserved         /* Reserved field to pad out to 64 bit boundary */
    MQCHAR28   ApplName         /* Name set by the application to identify the connection to
    the queue manager */
    MQBYTE4    Reserved2        /* Reserved field to pad out to 64 bit boundary */
    MQLONG     BalanceParmsOffset /* Offset of the MQBMO structure */
}
```



```

    PMQBMO      BalanceParmsPtr    /* Address of the location of the MQBMO structure */
};

```

## Déclaration COBOL pour MQCNO

```

** MQCNO structure
10 MQCNO.
** Structure identifier
15 MQCNO-STRUCID PIC X(4).
** Structure version number
15 MQCNO-VERSION PIC S9(9) BINARY.
** Options that control the action of MQCONN
15 MQCNO-OPTIONS PIC S9(9) BINARY.
** Offset of MQCD structure for client connection
15 MQCNO-CLIENTCONNOFFSET PIC S9(9) BINARY.
** Address of MQCD structure for client connection
15 MQCNO-CLIENTCONNPTR POINTER.
** Queue manager connection tag
15 MQCNO-CONNTAG PIC X(128).
** Address of MQSCO structure for client connection
15 MQCNO-SSLCONFIGPTR POINTER.
** Offset of MQSCO structure for client connection
15 MQCNO-SSLCONFIGOFFSET PIC S9(9) BINARY.
** Unique connection identifier
15 MQCNO-CONNECTIONID PIC X(24).
** Offset of MQCSP structure for security parameters
15 MQCNO-SECURITYPARMSOFFSET PIC S9(9) BINARY.
** Address of MQCSP structure for security parameters
15 MQCNO-SECURITYPARMSPTR POINTER.
** Length of string identified by CCDTURLLENGTH or CCDURLPTR
15 MQCNO-CCDTURLLENGTH
** Pointer to a string which contains a URL, to identify the location of the client
connection channel
15 MQCNO-CCDTURLPTR
** Address of string which contains a URL that identifies the location of the client
connection channel table
15 MQCNO-CCDTURLOFFSET
** Reserved field to pad to 64 bit boundary
15 MQCNO-RESERVED
** Name set by the application to identify the connection to the queue manager
15 MQCNO-APPLNAME
** Reserved field to pad to 64 bit boundary
15 MQCNO-RESERVED2
** Address of the MQBMO structure
15 MQCNO-BALANCEPARMSOFFSET
** Pointer to the MQBMO structure
15 MQCNO-BALANCEPARMSPTR

```

## Déclaration PL/I pour MQCNO

```

dcl
1 MQCNO based,
3 StrucId          char(4),          /* Structure identifier */
3 Version          fixed bin(31),   /* Structure version number */
3 Options          fixed bin(31),   /* Options that control the action
of MQCONN */
3 ClientConnOffset fixed bin(31),   /* Offset of MQCD structure for
client connection */
3 ClientConnPtr   pointer,          /* Address of MQCD structure for
client connection */
3 ConnTag         char(128),        /* Queue manager connection tag */
3 SSLConfigPtr    pointer,          /* Address of MQSCO structure for
client connection */
3 SSLConfigOffset fixed bin(31),   /* Offset of MQSCO structure for
client connection */
3 ConnectionId    char(24),         /* Unique connection identifier
3 SecurityParmsOffset fixed bin(31) /* Offset of MQCSP structure for
security parameters */
3 SecurityParmsPtr pointer,          /* Address of MQCSP structure for
security parameters */
3 CCDUrlLength    fixed bin(31)     /* Length of string identified by CCDUrlPtr
or CCDUrlOffset */
3 CCDUrlOffset    fixed bin(31)     /* Offset in bytes to URL of client connection channel */
3 CCDUrlPtr       pointer           /* Pointer to string containing URL */
3 Reserved        char(4)           /* Reserved field to pad out to 64 bit boundary */
3 ApplName        char(28)          /* Name set by the application to identify the
connection to

```

```

3 Reserved2          char(4)          /* the queue manager */
3 BalanceParmsOffset fixed bin(31) /* Reserved field to pad out to 64 bit boundary */
3 BalanceParmsPtr    pointer          /* Offset of the MQBMO structure */
3 BalanceParmsPtr    pointer          /* Address of the MQBMO structure */

```

## Déclaration High Level Assembler pour MQCNO

```

MQCNO                DSECT
MQCNO_STRUCID        DS    CL4      Structure identifier
MQCNO_VERSION        DS    F        Structure version number
MQCNO_OPTIONS        DS    F        Options that control the action of
*                    MQCONN
MQCNO_CLIENTCONNOFFSET DS    F      Offset of MQCD structure for client
*                    connection
MQCNO_CLIENTCONNPTR  DS    F        Address of MQCD structure for client
*                    connection
MQCNO_CONNTAG        DS    XL128     Queue manager connection tag
MQCNO_CONNECTIONID   DS    XL24     Unique connection identifier
MQCNO_SSLCONFIGOFFSET DS    F        Offset of MQCSP structure for security
*                    parameters
MQCNO_SSLCONFIGPTR    DS    F        Address of MQCSP structure for security
*                    parameters
MQCNO_LENGTH        EQU    *-MQCNO
                    ORG    MQCNO
MQCNO_AREA          DS    CL(MQCNO_LENGTH)
MQCNO_CCDTURLLENGTH DS    F        Length of string identified by CCDTURLPTR or
*                    CCDTURLOFFSET
MQCNO_CCDTURLOFFSET DS    F        Offset in bytes to URL of client connection channel
MQCNO_CCDTURLPTR     DS    F        Pointer to string containing URL
RESERVED            DS    XL4       Reserved field to pad out to 64 bit boundary
APPLNAME           DS    CL28      Name set by the application to identify the connection to
*                    the queue manager
RESERVED2          DS    XL4       Reserved field to pad out to 64 bit boundary
MQCNO_BALANCEPARMSOFFSET DS    F    Offset of the MQBMO structure
MQCNO_BALANCEPARMSPTR DS    F      Address of the MQBMO structure

```

## Déclaration Visual Basic pour MQCNO

```

Type MQCNO
  StrucId          As String*4      'Structure identifier'
  Version          As Long          'Structure version number'
  Options          As Long          'Options that control the action of'
  Options          As Long          'MQCONN'
  ClientConnOffset As Long          'Offset of MQCD structure for client'
  ClientConnOffset As Long          'connection'
  ClientConnPtr    As MQPTR         'Address of MQCD structure for client'
  ClientConnPtr    As MQPTR         'connection'
  ConnTag          As MQBYTE128     'Queue manager connection tag'
  SSLConfigPtr     As MQPTR         'Address of MQSCO structure for client'
  SSLConfigPtr     As MQPTR         'connection'
  SSLConfigOffset  As Long          'Offset of MQSCO structure for client'
  SSLConfigOffset  As Long          'connection'
  ConnectionId     As MQBYTE24     'Unique connection identifier'
  SecurityParmsOffset As Long       'Offset of MQCSP structure for security'
  SecurityParmsOffset As Long       'parameters'
  SecurityParmsPtr As MQPTR         'Address of MQCSP structure for security'
  SecurityParmsPtr As MQPTR         'parameters'
  CCDTURLLength    As Long          'Length of string identified by CCDTURLPtr'
  CCDTURLLength    As Long          'or CCDTURLOffset'
  CCDTURLOffset    As Long          'Offset in bytes to URL of client connection channel'
  CCDTURLPtr       As MQPTR         'Pointer to string containing URL'
  Reserved         As MQBYTE4      'Reserved field to pad out to 64 bit boundary'
  ApplName         As String*28    'Name set by the application to identify the connection to'
  ApplName         As String*28    'the queue manager'
  Reserved2        As MQBYTE4      'Reserved field to pad out to 64 bit boundary'
  BalanceParmsOffset As Long        'Offset in bytes to MQBNO structure'
  BalanceParmsPtr  As MQPTR         'Address of MQBNO structure'
End Type

```

## Tâches associées

### Utilisation de MQCONN

### ***StrucId (MQCHAR4) pour MQCNO***

Il s'agit de l'identificateur de structure de la structure des options de connexion. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQCNO\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUC\_MQCNO**

Identificateur de la structure des options de connexion.

Pour le langage de programmation C, la constante MQCNO\_STRUC\_ID\_ARRAY est également définie. Cette constante a la même valeur que MQCNO\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### ***Version (MQLONG) pour MQCNO***

La version est toujours une zone d'entrée. Sa valeur initiale est MQCNO\_VERSION\_1.

Il doit s'agir de l'une des valeurs suivantes :

#### **MQCNO\_VERSION\_1**

Structure d'options de connexion Version-1 .

#### **MQCNO\_VERSION\_2**

Structure d'options de connexion Version-2 .

#### **MQCNO\_VERSION\_3**

Structure d'options de connexion Version-3 .

#### **MQCNO\_VERSION\_4**

Structure des options de connexion Version-4 .

#### **MQCNO\_VERSION\_5**

Structure d'options de connexion Version-5 .

#### **MQCNO\_VERSION\_6**

Structure d'options de connexion Version-6 .

#### **MQCNO\_VERSION\_7**

Structure des options de connexion Version-7 .

#### **V 9.3.0 MQCNO\_VERSION\_8**

Structure d'options de connexion Version-8 .

Les zones qui existent uniquement dans les versions plus récentes de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

#### **MQCNO\_CURRENT\_VERSION**

Version actuelle de la structure des options de connexion.

### ***Options (MQLONG) pour MQCNO***

Options qui contrôlent l'action de MQCONN.

## **Options de comptabilité**

Les options suivantes contrôlent le type de comptabilité si l'attribut de gestionnaire de files d'attente **AccountingConnOverride** est défini sur MQMON\_ENABLED:

#### **MQCNO\_ACCOUNTING\_MQI\_ENABLED**

Lorsque la collecte des données de surveillance est désactivée dans la définition du gestionnaire de files d'attente en définissant l'attribut **MQIAccounting** sur MQMON\_OFF, la définition de cet indicateur active la collecte des données de comptabilité MQI.

### **MQCNO\_ACCOUNTING\_MQI\_DISABLED**

Lorsque la collecte des données de surveillance est désactivée dans la définition du gestionnaire de files d'attente en définissant l'attribut **MQIAccounting** sur MQMON\_OFF, la définition de cet indicateur arrête la collecte des données de comptabilité MQI.

### **MQCNO\_ACCOUNTING\_Q\_ENABLED**

Lorsque la collecte des données de comptabilité des files d'attente est désactivée dans la définition de gestionnaire de files d'attente en définissant l'attribut **MQIAccounting** sur MQMON\_OFF, la définition de cet indicateur active la collecte des données de comptabilité pour les files d'attente qui spécifient un gestionnaire de files d'attente dans la zone *MQIAccounting* de leur définition de file d'attente.

### **MQCNO\_ACCOUNTING\_Q\_DISABLED**

Lorsque la collecte des données de comptabilité des files d'attente est désactivée dans la définition de gestionnaire de files d'attente en définissant l'attribut **MQIAccounting** sur MQMON\_OFF, cette option désactive la collecte des données de comptabilité pour les files d'attente qui spécifient un gestionnaire de files d'attente dans la zone *MQIAccounting* de leur définition de file d'attente.

Si aucun de ces indicateurs n'est défini, la prise en compte de la connexion est celle définie dans les attributs du gestionnaire de files d'attente.

## **Options de liaison**

Les options suivantes contrôlent le type de liaison IBM MQ à utiliser. Spécifiez une seule de ces options:

### **MQCNO\_STANDARD\_BINDING**

L'application et l'agent du gestionnaire de files d'attente local (le composant qui gère les opérations de mise en file d'attente) s'exécutent dans des unités d'exécution distinctes (généralement, dans des processus distincts). Cette disposition maintient l'intégrité du gestionnaire de files d'attente, c'est-à-dire qu'elle protège le gestionnaire de files d'attente des programmes errants.

Si le gestionnaire de files d'attente prend en charge plusieurs types de liaison et que vous définissez MQCNO\_STANDARD\_BINDING, il utilise l'attribut **DefaultBindType** dans la section Connection du fichier `qm.ini` pour sélectionner le type de liaison réel. Si cette section n'est pas définie, si la valeur ne peut pas être utilisée ou si elle n'est pas appropriée pour l'application, le gestionnaire de files d'attente sélectionne un type de liaison approprié. Le gestionnaire de files d'attente définit le type de liaison réel utilisé dans les options de connexion.

Utilisez MQCNO\_STANDARD\_BINDING dans les situations où l'application n'a peut-être pas été entièrement testée ou n'est pas fiable ou n'est pas fiable. MQCNO\_STANDARD\_BINDING est la valeur par défaut.

Cette option est prise en charge dans tous les environnements.

Si vous établissez une liaison à la bibliothèque `mqm`, une connexion serveur standard utilisant le type de liaison par défaut est tentée en premier. Si le chargement de la bibliothèque du serveur sous-jacent a échoué, une connexion client est tentée à la place.

- Pour modifier le comportement de MQCONN (ou de MQCONNX si MQCNO\_STANDARD\_BINDING est spécifié), définissez la variable d'environnement MQ\_CONNECT\_TYPE sur l'une des options suivantes. Notez qu'il existe une exception: si MQCNO\_FASTPATH\_BINDING est spécifié avec MQ\_CONNECT\_TYPE défini sur LOCAL ou STANDARD, les connexions fastpath peuvent être rétro-migrées par l'administrateur sans modification associée à l'application.

Valeur	Explication
CLIENT	Une connexion client uniquement est tentée.
Fastpath	Cette valeur était prise en charge dans les éditions précédentes, mais elle est désormais ignorée si elle est spécifiée.

Tableau 480. Valeurs de MQ\_CONNECT\_TYPE qui modifient le comportement de MQCONN ou MQCONNX (suite)

Valeur	Explication
LOCAL	Une connexion serveur uniquement est tentée. Les connexions Fastpath sont rétromigrées vers une connexion serveur standard.
STANDARD	Prise en charge pour la compatibilité avec les éditions précédentes. Cette valeur est désormais traitée comme étant LOCAL.

- Si la variable d'environnement MQ\_CONNECT\_TYPE n'est pas définie lorsque MQCONNX est appelé, une connexion serveur standard utilisant le type de liaison par défaut est tentée. Si le chargement de la bibliothèque du serveur échoue, une connexion client est tentée.


### MQCNO\_FASTPATH\_BINDING


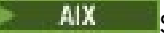
L'application et l'agent du gestionnaire de files d'attente local font partie de la même unité d'exécution. Cela contraste avec la méthode classique de liaison, où l'application et l'agent du gestionnaire de files d'attente local s'exécutent dans des unités d'exécution distinctes.

MQCNO\_FASTPATH\_BINDING est ignoré si le gestionnaire de files d'attente ne prend pas en charge ce type de liaison ; le traitement se poursuit comme si l'option n'avait pas été spécifiée.


MQCNO\_FASTPATH\_BINDING peut être avantageux dans les situations où plusieurs processus consomment plus de ressources que la ressource globale utilisée par l'application. Une application qui utilise la liaison fastpath est appelée *application sécurisée*.

Prenez en compte les points importants suivants lorsque vous décidez d'utiliser la liaison de raccourci:

- L'utilisation de l'option MQCNO\_FASTPATH\_BINDING n'empêche pas une application de modifier ou d'altérer des messages et d'autres zones de données appartenant au gestionnaire de files d'attente. Utilisez cette option uniquement dans les situations où vous avez intégralement évalué ces problèmes.
- L'application ne doit pas utiliser de signaux asynchrones ou d'interruptions de temporisateur (telles que `sigkill`) avec MQCNO\_FASTPATH\_BINDING. Il existe également des restrictions sur l'utilisation des segments de mémoire partagée.
- L'application doit utiliser l'appel MQDISC pour se déconnecter du gestionnaire de files d'attente.
- L'application doit se terminer avant que vous n'arrêtiez le gestionnaire de files d'attente à l'aide de la commande `endmqm`.
-  Sous IBM i, le travail doit s'exécuter sous un profil utilisateur appartenant au groupe QMQMADM. En outre, le programme ne doit pas s'arrêter anormalement, sinon des résultats imprévisibles peuvent se produire.

-   Sous AIX and Linux, l'identificateur d'utilisateur `mqm` doit être l'identificateur d'utilisateur effectif et l'identificateur de groupe `mqm` doit être l'identificateur de groupe effectif. Pour que l'application s'exécute de cette manière, configurez le programme de sorte qu'il appartienne à l'identificateur d'utilisateur `mqm` et à l'identificateur de groupe `mqm`, puis définissez les bits d'autorisation `setuid` et `setgid` sur le programme.

IBM MQ Object Authority Manager (OAM) utilise toujours l'ID utilisateur réel pour la vérification des droits d'accès.

-  Sous Windows, le programme doit être membre du groupe `mqm`. La liaison Fastpath n'est pas prise en charge pour les applications 64 bits.

L'option MQCNO\_FASTPATH\_BINDING est prise en charge dans les environnements suivants:

-  AIX

- **IBM i** IBM i
- **Linux** Linux
- **Windows** Windows

▶ **z/OS** Sous z/OS, l'option est acceptée mais ignorée.

Pour plus d'informations sur les implications de l'utilisation d'applications de confiance, voir [Restrictions pour les applications de confiance](#).

### **MQCNO\_LIEN\_PARTAGE**

Avec MQCNO\_SHARED\_BINDING, l'application et l'agent du gestionnaire de files d'attente local partagent certaines ressources. MQCNO\_SHARED\_BINDING est ignoré si le gestionnaire de files d'attente ne prend pas en charge ce type de liaison. Le traitement se poursuit comme si l'option n'avait pas été spécifiée.

### **MQCNO\_LIEN\_ISOLÉ\_LIAISON**

Dans ce cas, le processus d'application et l'agent du gestionnaire de files d'attente local sont isolés les uns des autres car ils ne partagent pas de ressources. MQCNO\_ISOLATED\_BINDING est ignoré si le gestionnaire de files d'attente ne prend pas en charge ce type de liaison. Le traitement se poursuit comme si l'option n'avait pas été spécifiée.

### **LIEN\_CLIENT\_MQCNO\_BINDING**

Spécifiez cette option pour que l'application tente une connexion client uniquement. Cette option présente les limitations suivantes:

- ▶ **z/OS** MQCNO\_CLIENT\_BINDING est ignoré sous z/OS.
- MQCNO\_CLIENT\_BINDING est rejeté avec MQRC\_OPTIONS\_ERROR s'il est spécifié avec une option de liaison MQCNO autre que MQCNO\_STANDARD\_BINDING.
- MQCNO\_CLIENT\_BINDING n'est pas disponible pour Java ou .NET car ils disposent de leurs propres mécanismes pour choisir le type de liaison.

### **MQCNO\_LIEN\_LOCAL\_LOCAL**

Indiquez cette option pour que l'application tente de se connecter au serveur. Si MQCNO\_FASTPATH\_BINDING, MQCNO\_ISOLATED\_BINDING ou MQCNO\_SHARED\_BINDING est également spécifié, la connexion est de ce type et est documentée dans cette section. Sinon, une connexion serveur standard est tentée à l'aide du type de liaison par défaut. MQCNO\_LOCAL\_BINDING présente les limitations suivantes:

- ▶ **z/OS** MQCNO\_LOCAL\_BINDING est ignoré sous z/OS.
- MQCNO\_LOCAL\_BINDING est rejeté avec MQRC\_OPTIONS\_ERROR s'il est spécifié avec une option de reconnexion MQCNO autre que MQCNO\_RECONNECT\_AS\_DEF.
- MQCNO\_LOCAL\_BINDING n'est pas disponible pour Java ou .NET car ils disposent de leurs propres mécanismes pour choisir le type de liaison.

Sur les plateformes suivantes, vous pouvez utiliser la variable d'environnement MQ\_CONNECT\_TYPE avec le type de liaison spécifié par la zone Options pour contrôler le type de liaison utilisé.

- **AIX** AIX
- **Linux** Linux
- **Windows** Windows

Si vous spécifiez cette variable d'environnement, elle doit avoir la valeur FASTPATH ou STANDARD ; s'il a une valeur différente, il est ignoré. La valeur de la variable d'environnement est sensible à la casse ; voir [Variable d'environnement MQCONNX](#) pour plus d'informations.

La variable d'environnement et la zone *Options* interagissent comme suit:

- Si vous omettez la variable d'environnement ou que vous lui attribuez une valeur qui n'est pas prise en charge, l'utilisation de la liaison raccourci est déterminée uniquement par la zone *Options*.
- Si vous attribuez à la variable d'environnement une valeur prise en charge, la liaison fastpath est utilisée uniquement si la variable d'environnement et la zone *Options* spécifient la liaison fastpath.

## Options de balise de connexion

Ces options sont prises en charge uniquement lors de la connexion à un gestionnaire de files d'attente z/OS et contrôlent l'utilisation de la balise de connexion ConnTag. Vous ne pouvez spécifier qu'une seule de ces options.

L'implémentation précise des balises de connexion diffère entre IBM MQ for z/OS et IBM MQ for Multiplatforms:

- ▶ **z/OS** Les options suivantes, à l'exception de `MQCNO_GENERATE_CONN_TAG`, sont prises en charge uniquement lors de la connexion à un gestionnaire de files d'attente z/OS et contrôlent l'utilisation de la balise de connexion. Vous ne pouvez spécifier qu'une seule des options prises en charge.
- ▶ **ALW** `MQCNO_GENERATE_CONN_TAG` n'est pris en charge que sur les plateformes autres que z/OS.

### ▶ **ALW** `MQCNO_GENERATE_CONN_TAG`

Renvoie la balise de connexion que le gestionnaire de files d'attente a associée à cette connexion, dans la structure MQCNO de sortie.

La balise de connexion renvoyée sera identique pour toutes les connexions que le gestionnaire de files d'attente considère comme une instance d'application unique.

### ▶ **z/OS** `MQCNO_SERIALIZE_CONN_TAG_Q_MGR`

Cette option demande l'utilisation exclusive de la balise de connexion dans le gestionnaire de files d'attente local. Si la balise de connexion est déjà utilisée dans le gestionnaire de files d'attente local, l'appel MQCONNX échoue avec le code anomalie MQRC\_CONN\_TAG\_IN\_USE. Le résultat de l'appel n'est pas affecté par l'utilisation de la balise de connexion ailleurs dans le groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local.

### ▶ **z/OS** `MQCNO_SERIALIZE_CONN_TAG_QSG`

Cette option demande l'utilisation exclusive de la balise de connexion dans le groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local. Si la balise de connexion est déjà utilisée dans le groupe de partage de files d'attente, l'appel MQCONNX échoue avec le code anomalie MQRC\_CONN\_TAG\_IN\_USE.

### ▶ **z/OS** `MQCNO_RESTRICT_CONN_TAG_Q_MGR`

Cette option demande l'utilisation partagée de la balise de connexion dans le gestionnaire de files d'attente local. Si la balise de connexion est déjà utilisée dans le gestionnaire de files d'attente local, l'appel MQCONNX peut aboutir si l'application demandeuse s'exécute dans la même portée de traitement que l'utilisateur existant de la balise. Si cette condition n'est pas satisfaite, l'appel MQCONNX échoue avec le code anomalie MQRC\_CONN\_TAG\_IN\_USE. Le résultat de l'appel n'est pas affecté par l'utilisation de la balise de connexion ailleurs dans le groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local.

- Les applications doivent s'exécuter dans le même espace adresse MVS pour partager la balise de connexion. Si l'application utilisant la balise de connexion est une application client, MQCNO\_RESTRICT\_CONN\_TAG\_Q\_MGR n'est pas autorisé.

### MQCNO\_RESTRICT\_CONN\_TAG\_QSG

Cette option demande l'utilisation partagée de la balise de connexion dans le groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local. Si la balise de connexion est déjà utilisée dans le groupe de partage de files d'attente, l'appel MQCONNX peut aboutir à condition que l'application demandeuse s'exécute dans la même portée de traitement et soit connectée au même gestionnaire de files d'attente que l'utilisateur existant de la balise.

Si ces conditions ne sont pas satisfaites, l'appel MQCONNX échoue avec le code anomalie MQRC\_CONN\_TAG\_IN\_USE.

- Les applications doivent s'exécuter dans le même espace adresse MVS pour partager la balise de connexion. Si l'application utilisant la balise de connexion est une application client, MQCNO\_RESTRICT\_CONN\_TAG\_QSG n'est pas autorisé.

Si aucune de ces options n'est spécifiée, ConnTag n'est pas utilisé. Ces options ne sont pas valides si Version est inférieur à MQCNO\_VERSION\_3.

## Options de partage de descripteur

### Multi

Ces options sont prises en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows

Ils contrôlent le partage des descripteurs entre différentes unités d'exécution (unités de traitement parallèle) au sein d'un même processus. Vous ne pouvez spécifier qu'une seule des options suivantes:

#### MQCNO\_HANDLE\_SHARE\_NONE

Cette option indique que les descripteurs de connexion et d'objet ne peuvent être utilisés que par l'unité d'exécution à l'origine de l'allocation du descripteur (c'est-à-dire l'unité d'exécution qui a émis l'appel MQCONN, MQCONNX ou MQOPEN). Les descripteurs ne peuvent pas être utilisés par d'autres unités d'exécution appartenant au même processus.

#### MQCNO\_HANDLE\_SHARE\_BLOCK

Cette option indique que les descripteurs de connexion et d'objet alloués par une unité d'exécution d'un processus peuvent être utilisés par d'autres unités d'exécution appartenant au même processus. Toutefois, une seule unité d'exécution à la fois peut utiliser une poignée particulière, c'est-à-dire que seule l'utilisation en série d'une poignée est autorisée. Si une unité d'exécution tente d'utiliser un descripteur déjà utilisé par une autre unité d'exécution, l'appel se bloque (attend) jusqu'à ce que le descripteur soit disponible.

#### MQCNO\_HANDLE\_SHARE\_NO\_BLOCK

Identique à MQCNO\_HANDLE\_SHARE\_BLOCK, sauf que si le descripteur est utilisé par une autre unité d'exécution, l'appel se termine immédiatement avec MQCC\_FAILED et MQRC\_CALL\_IN\_PROGRESS au lieu de se bloquer jusqu'à ce que le descripteur soit disponible.


Une unité d'exécution peut avoir zéro ou un descripteur non partagé:



- Chaque appel MQCONN ou MQCONNX qui spécifie MQCNO\_HANDLE\_SHARE\_NONE renvoie un nouveau descripteur non partagé sur le premier appel, et le même descripteur non partagé sur le deuxième appel et les appels ultérieurs (en supposant qu'il n'y ait pas d'appel MQDISC). Le code anomalie est MQRC\_ALREADY\_CONNECTED pour le deuxième appel et les appels ultérieurs.
- Chaque appel MQCONNX qui spécifie MQCNO\_HANDLE\_SHARE\_BLOCK ou MQCNO\_HANDLE\_SHARE\_NO\_BLOCK renvoie un nouveau descripteur partagé sur chaque appel.

Les descripteurs d'objet héritent des mêmes propriétés de partage que le descripteur de connexion indiqué dans l'appel MQOPEN qui a créé le descripteur d'objet. En outre, les unités de travail héritent des mêmes propriétés de partage que le descripteur de connexion utilisé pour démarrer l'unité de travail ; si l'unité de travail est démarrée dans une unité d'exécution à l'aide d'un descripteur partagé, l'unité de travail peut être mise à jour dans une autre unité d'exécution à l'aide du même descripteur.

Si vous ne spécifiez pas d'option de partage de descripteur, la valeur par défaut est déterminée par l'environnement:

-  Dans l'environnement Microsoft Transaction Server (MTS), la valeur par défaut est MQCNO\_HANDLE\_SHARE\_BLOCK.
- Dans d'autres environnements, la valeur par défaut est la même que MQCNO\_HANDLE\_SHARE\_NONE.

## Options de reconnexion

Les options de reconnexion déterminent si une connexion peut être reconnectée. Seules les connexions client sont reconnectables.

### MQCNO\_RECONNECT\_AS\_DEF

L'option de reconnexion est résolue avec sa valeur par défaut. Si aucune valeur par défaut n'est définie, la valeur de cette option est DISABLED. La valeur de l'option est transmise au serveur et peut être interrogée par PCF et MQSC.

### MQCNO\_RECONNECT

L'application peut être reconnectée à n'importe quel gestionnaire de files d'attente cohérent avec la valeur du paramètre **QmgrName** de MQCONNX. Utilisez l'option MQCNO\_RECONNECT uniquement s'il n'y a pas d'affinité entre l'application client et le gestionnaire de files d'attente avec lequel elle a initialement établi une connexion. La valeur de l'option est transmise au serveur et peut être interrogée par PCF et MQSC.

### MQCNO\_RECONNECT\_DISABLED

L'application ne peut pas être reconnectée. La valeur de l'option n'est pas transmise au serveur.

### MQCNO\_RECONNECT\_Q\_MGR

L'application ne peut être reconnectée qu'au gestionnaire de files d'attente avec lequel elle s'est connectée à l'origine. Utilisez cette valeur si un client peut être reconnecté, mais qu'il existe une affinité entre l'application client et le gestionnaire de files d'attente avec lequel il a établi une connexion. Choisissez cette valeur si vous souhaitez qu'un client se reconnecte automatiquement à l'instance de secours d'un gestionnaire de files d'attente haute disponibilité. La valeur de l'option est transmise au serveur et peut être interrogée par PCF et MQSC.

Utilisez les options MQCNO\_RECONNECT, MQCNO\_RECONNECT\_DISABLED et MQCNO\_RECONNECT\_Q\_MGR uniquement pour les connexions client. Si les options sont utilisées pour une connexion de liaison, MQCONNX échoue avec le code achèvement MQCC\_FAILED et le code anomalie MQRC\_OPTIONS\_ERROR. La reconnexion automatique du client n'est pas prise en charge par IBM MQ classes for Java

## Options de partage de conversation

Les options suivantes s'appliquent uniquement aux connexions client TCP/IP. Pour les canaux SNA, SPX et NetBios, ces valeurs sont ignorées et le canal s'exécute comme dans les versions précédentes du produit

### **MQCNO\_NO\_CONV\_SHARING**

Cette option n'autorise pas le partage de conversation.

Vous pouvez utiliser MQCNO\_NO\_NO\_CONV\_SHARING dans des situations où les conversations sont fortement chargées et, par conséquent, où un conflit est possible à l'extrémité de connexion serveur de l'instance de canal sur laquelle les conversations partagées existent. MQCNO\_NO\_CONV\_SHARING se comporte comme SHARECNV (1) lorsqu'il est connecté à un canal qui prend en charge le partage des conversations et SHARECNV (0) lorsqu'il est connecté à un canal qui ne prend pas en charge le partage des conversations.

### **MQCNO\_TOUS\_CONVS\_SHARE**

Cette option permet le partage des conversations ; l'application ne limite pas le nombre de connexions sur l'instance de canal. Il s'agit de la valeur par défaut.

Si l'application indique que l'instance de canal peut être partagée, mais que la définition *SharingConversations* (SHARECNV) sur l'extrémité de connexion serveur du canal est définie sur une, aucun partage n'a lieu et aucun avertissement n'est envoyé à l'application.

De même, si l'application indique que le partage est autorisé mais que la définition *SharingConversations* de la connexion serveur est définie sur zéro, aucun avertissement n'est émis et l'application présente le même comportement qu'un client dans les versions antérieures à IBM WebSphere MQ 7.0; le paramètre d'application relatif au partage des conversations est ignoré.

MQCNO\_NO\_CONV\_SHARING et MQCNO\_ALL\_CONVS\_SHARE s'excluent mutuellement. Si les deux options sont spécifiées sur une connexion particulière, la connexion est rejetée avec le code anomalie MQRC\_OPTIONS\_ERROR.

## **Options de définition de canal**

Les options suivantes contrôlent l'utilisation de la structure de définition de canal transmise dans le MQCNO:

### **MQCNO\_CD\_FOR\_OUTPUT\_ONLY**

Cette option permet d'utiliser la structure de définition de canal dans le MQCNO uniquement pour renvoyer le nom de canal utilisé lors d'un appel MQCONNX réussi.

Si une structure de définition de canal valide n'est pas fournie, l'appel échoue avec le code anomalie MQRC\_CD\_ERROR.

Si l'application n'est pas exécutée en tant que client, l'option est ignorée.

Le nom de canal renvoyé peut être utilisé lors d'un appel MQCONNX ultérieur à l'aide de l'option MQCNO\_USE\_CD\_SELECTION pour se reconnecter à l'aide de la même définition de canal. Cela peut être utile lorsqu'il existe plusieurs définitions de canal applicables dans la table des canaux client.

### **MQCNO\_USE\_CD\_SELECTION**

Cette option permet à l'appel MQCONNX de se connecter à l'aide du nom de canal contenu dans la structure de définition de canal transmise dans le MQCNO.

Si la variable d'environnement MQSERVER est définie, la définition de canal qu'elle définit est utilisée. Si MQSERVER n'est pas défini, la table de canaux client est utilisée.

Si aucune définition de canal avec un nom de canal et un nom de gestionnaire de files d'attente correspondants n'est trouvée, l'appel échoue avec le code anomalie MQRC\_Q\_MGR\_NAME\_ERROR.

Si une structure de définition de canal valide n'est pas fournie, l'appel échoue avec le code anomalie MQRC\_CD\_ERROR.

Si l'application n'est pas exécutée en tant que client, l'option est ignorée.

## Option par défaut

Si vous n'avez besoin d'aucune des options décrites ci-dessus, vous pouvez utiliser l'option suivante:

### **MQCNO\_AUCUN**

Aucune option n'est spécifiée.

Utilisez MQCNO\_NONE pour faciliter la documentation du programme. Il n'est pas prévu que cette option soit utilisée avec une autre option MQCNO\_\*, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

### **ClientConnDécalage (MQLONG) pour MQCNO**

ClientConnLe décalage est le décalage en octets d'une structure de définition de canal MQCD à partir du début de la structure MQCNO. Le décalage peut être positif ou négatif. Cette zone est une zone d'entrée avec une valeur initiale de 0.

Utilisez *ClientConnOffset* uniquement lorsque l'application émettrice de l'appel MQCONNX s'exécute en tant que IBM MQ MQI client. Pour plus d'informations sur l'utilisation de cette zone, voir la description de la zone *ClientConnPtr*.

Cette zone est ignorée si *Version* est inférieur à MQCNO\_VERSION\_2.

### **ClientConnPtr (MQPTR) pour MQCNO**

ClientConnPtr est une zone d'entrée. Sa valeur initiale est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire.

Utilisez *ClientConnOffset* et *ClientConnPtr* uniquement lorsque l'application émettant l'appel MQCONNX s'exécute en tant que IBM MQ MQI client. En spécifiant l'une ou l'autre de ces zones, l'application peut contrôler la définition du canal de connexion client en fournissant une structure de définition de canal MQCD qui contient les valeurs requises.

Si l'application s'exécute en tant que IBM MQ MQI client, mais ne fournit pas de structure MQCD, la variable d'environnement MQSERVER est utilisée pour sélectionner la définition de canal. Si MQSERVER n'est pas défini, la table de canaux client est utilisée.

Si l'application n'est pas exécutée en tant que IBM MQ MQI client, *ClientConnOffset* et *ClientConnPtr* sont ignorés.

Si l'application fournit une structure MQCD, définissez les zones répertoriées sur les valeurs requises ; les autres zones de MQCD sont ignorées. Vous pouvez compléter les chaînes de caractères avec des blancs à la longueur de la zone ou les terminer par un caractère NULL. Pour plus d'informations sur les zones de la structure MQCD, voir «Zones», à la page 1554.

Tableau 481. Zones dans MQCD

<b>Zone dans MQCD</b>	<b>Valeur</b>
<i>ChannelName</i>	Nom de canal.
<i>Version</i>	Numéro de version de la structure. Ne doit pas être inférieur à MQCD_VERSION_7.
<i>TransportType</i>	Tout type de transport pris en charge.
<i>ModeName</i>	Nom de mode LU 6.2 .
<i>TpName</i>	Nom du programme de transaction LU 6.2 .
<i>SecurityExit</i>	Nom de l'exit de sécurité de canal.

Tableau 481. Zones dans MQCD (suite)

<b>Zone dans MQCD</b>	<b>Valeur</b>
<i>SendExit</i>	Nom de l'exit d'émission de canal.
<i>ReceiveExit</i>	Nom de l'exit de réception de canal.
<i>MaxMsgLength</i>	Longueur maximale en octets des messages pouvant être envoyés via le canal de connexion client.
<i>SecurityUserData</i>	Données utilisateur pour l'exit de sécurité.
<i>SendUserData</i>	Données utilisateur pour l'exit d'émission.
<i>ReceiveUserData</i>	Données utilisateur de l'exit de réception.
<i>UserIdentifier</i>	ID utilisateur à utiliser pour établir une session LU 6.2 .
<i>Password</i>	Mot de passe à utiliser pour établir une session LU 6.2 .
<i>ConnectionName</i>	nom de la connexion.
<i>HeartbeatInterval</i>	Durée en secondes entre les flux de pulsations.
<i>StrucLength</i>	Longueur de la structure MQCD.
<i>ExitNameLength</i>	Longueur des noms d'exit traités par <i>SendExitPtr</i> et <i>ReceiveExitPtr</i> . Doit être supérieur à zéro si <i>SendExitPtr</i> ou <i>ReceiveExitPtr</i> est défini sur une valeur qui n'est pas le pointeur null.
<i>ExitDataLength</i>	Longueur des données de sortie traitées par <i>SendUserDataPtr</i> et <i>ReceiveUserDataPtr</i> . Doit être supérieur à zéro si <i>SendUserDataPtr</i> ou <i>ReceiveUserDataPtr</i> est défini sur une valeur qui n'est pas le pointeur null.
<i>SendExitsDefined</i>	Nombre d'exits d'émission traités par <i>SendExitPtr</i> . Si la valeur est zéro, <i>SendExit</i> et <i>SendUserData</i> fournissent le nom et les données de l'exit. Si la valeur est supérieure à zéro, <i>SendExitPtr</i> et <i>SendUserDataPtr</i> fournissent les noms et les données des exits, et <i>SendExit</i> et <i>SendUserData</i> doivent être vides.
<i>ReceiveExitsDefined</i>	Nombre d'exits de réception traités par <i>ReceiveExitPtr</i> . Si la valeur est zéro, <i>ReceiveExit</i> et <i>ReceiveUserData</i> fournissent le nom et les données de l'exit. Si la valeur est supérieure à zéro, <i>ReceiveExitPtr</i> et <i>ReceiveUserDataPtr</i> fournissent les noms et les données des exits, et <i>ReceiveExit</i> et <i>ReceiveUserData</i> doivent être vides.
<i>SendExitPtr</i>	Adresse du nom du premier exit d'émission.
<i>SendUserDataPtr</i>	Adresse des données pour le premier exit d'émission.
<i>ReceiveExitPtr</i>	Adresse du nom du premier exit de réception.
<i>ReceiveUserDataPtr</i>	Adresse des données pour le premier exit de réception.
<i>LongRemoteUserIdLength</i>	Longueur de l'ID utilisateur éloigné long.
<i>LongRemoteUserIdPtr</i>	Adresse de l'ID utilisateur distant long.
<i>RemoteSecurityId</i>	Identificateur de sécurité distant.
<i>SSLCipherSpec</i>	TLS CipherSpec.
<i>SSLPeerNamePtr</i>	Adresse du nom d'homologue TLS.
<i>SSLPeerNameLength</i>	Longueur du nom d'homologue TLS.
<i>KeepAliveInterval</i>	Valeur transmise à la pile de communications pour la temporisation du signal de présence pour le canal

Tableau 481. Zones dans MQCD (suite)

Zone dans MQCD	Valeur
<i>LocalAddress</i>	Adresse de communication locale, y compris l'adresse IP de l'adaptateur de réseau local à utiliser, et plage de ports à utiliser pour les connexions sortantes.

Fournissez la structure de définition de canal de l'une des deux manières suivantes:

- En utilisant la zone de décalage *ClientConnOffset*

Dans ce cas, l'application doit déclarer une structure composée contenant un MQCNO suivi de la structure de définition de canal MQCD et définir *ClientConnOffset* sur le décalage de la structure de définition de canal à partir du début de MQCNO. Vérifiez que ce décalage est correct. *ClientConnPtr* doit être défini sur le pointeur null ou sur les octets null.

Utilisez *ClientConnOffset* pour les langages de programmation qui ne prennent pas en charge le type de données de pointeur ou qui implémentent le type de données de pointeur d'une manière qui n'est pas portable dans des environnements différents (par exemple, le langage de programmation COBOL).

Pour le langage de programmation Visual Basic, une structure composée appelée MQCNOCD est fournie dans le fichier d'en-tête CMQXB.BAS; cette structure contient une structure MQCNO suivie d'une structure MQCD. Initialisez MQCNOCD en appelant la sous-routine MQCNOCD\_DEFAULTS. MQCNOCD est utilisé avec le Variante MQCONNXAny de l'appel MQCONNX ; voir la description de l'appel MQCONNX pour plus de détails.

- A l'aide de la zone de pointeur *ClientConnPtr*

Dans ce cas, l'application peut déclarer la structure de définition de canal séparément de la structure MQCNO et définir *ClientConnPtr* sur l'adresse de la structure de définition de canal. Définissez *ClientConnOffset* sur zéro.

Utilisez *ClientConnPtr* pour les langages de programmation qui prennent en charge le type de données de pointeur d'une manière portable dans différents environnements (par exemple, le langage de programmation C).

Dans le langage de programmation C, vous pouvez utiliser la variable de macro MQCD\_CLIENT\_CONN\_DEFAULT pour fournir des valeurs initiales pour la structure qui conviennent mieux à l'utilisation sur l'appel MQCONNX que les valeurs initiales fournies par MQCD\_DEFAULT.

Quelle que soit la technique choisie, vous ne pouvez utiliser qu'une seule des options *ClientConnOffset* et *ClientConnPtr* ; l'appel échoue avec le code anomalie MQRC\_CLIENT\_CONN\_ERROR si les deux sont différents de zéro.

Une fois l'appel MQCONNX terminé, la structure MQCD n'est plus référencée.

Cette zone est ignorée si *Version* est inférieur à MQCNO\_VERSION\_2.

**Remarque :** Sur les plateformes où le langage de programmation ne prend pas en charge le type de données pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée, la valeur initiale étant la chaîne d'octets tout-null.

### **ConnTag (MQBYTE128) pour MQCNO sur Multiplatforms**

Une balise de connexion est conceptuellement similaire à un identificateur de connexion, mais peut s'étendre sur plusieurs connexions associées, en les identifiant comme une seule instance d'application. Sur Multiplatforms, la balise de connexion est générée par le gestionnaire de files d'attente lors de la connexion.

Pour plus d'informations, voir [Identificateur de connexion](#) et [instance d'application](#).

Les balises de connexion générées sont semi lisibles par l'utilisateur. En d'autres termes, ils peuvent être affichés et filtrés dans MQSC comme s'ils étaient des chaînes du jeu de caractères local. Les connexions

associées à IBM MQ sont automatiquement affectées à la même balise de connexion. Cette affectation est particulièrement importante pour l' équilibre des applications.

La balise de connexion générée est visible de trois manières:

- Dans la structure MQCNO de sortie sur un appel MQCONN, lorsque MQCNO\_GENERATE\_CONN\_TAG est spécifié.
- Dans la sortie de DISPLAY CONN (ou des équivalents programmatiques).
- Dans la sortie de DISPLAY APSTATUS (ou équivalents).

La balise cesse d'être valide lorsque l'application s'arrête ou émet l'appel MQDISC.

### Référence associée

«ConnTag (MQBYTE128) pour MQCNO sur IBM MQ for z/OS», à la page 342

Une balise de connexion est conceptuellement similaire à un identificateur de connexion, mais peut s'étendre sur plusieurs connexions associées, en les identifiant comme une seule instance d'application. Sous IBM MQ for z/OS, la balise de connexion est une zone d'entrée, fournie par l'application et utilisée conjointement avec les options MQCNO\_\*\_CONN\_TAG pour sérialiser les connexions à partir de cette instance d'application

### **ConnTag (MQBYTE128) pour MQCNO sur IBM MQ for z/OS**

Une balise de connexion est conceptuellement similaire à un identificateur de connexion, mais peut s'étendre sur plusieurs connexions associées, en les identifiant comme une seule instance d'application. Sous IBM MQ for z/OS, la balise de connexion est une zone d'entrée, fournie par l'application et utilisée conjointement avec les options MQCNO\_\*\_CONN\_TAG pour sérialiser les connexions à partir de cette instance d'application

Lorsque plusieurs instances d'une application sont destinées à être connectées simultanément, elles doivent chacune fournir une valeur unique pour cette zone. Pour plus de détails, voir les descriptions de ces options de balise de connexion .

### Remarques :

- Sous IBM MQ for z/OS, il n'est pas possible de déterminer administrativement la balise de connexion associée à une application lors de l'exécution.
- Les valeurs de balise de connexion commençant par MQ en majuscules, en minuscules ou en casse mixte en ASCII ou EBCDIC sont réservées aux produits IBM . N'utilisez pas de valeurs de balise de connexion commençant par ces lettres.

Utilisez la valeur spéciale suivante si vous n'avez besoin d'aucune balise:


### **MQCT\_AUCUN**

La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQCT\_NONE\_ARRAY est également définie ; cette constante a la même valeur que MQCT\_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La zone ConnTag est utilisée lors de la connexion à un gestionnaire de files d'attente z/OS .

La longueur de cette zone est indiquée par MQ\_CONN\_TAG\_LENGTH. Cette zone est ignorée si *Version* est inférieur à MQCNO\_VERSION\_3.

 Pour plus d'informations sur l'utilisation de la balise de connexion sur IBM MQ for Multiplatforms, voir «ConnTag (MQBYTE128) pour MQCNO sur Multiplatforms», à la page 341 .

### **SSLConfigPtr (PMQSCO) pour MQCNO**

SSLConfigPtr est une zone d'entrée. Sa valeur initiale est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire.

Utilisez *SSLConfigPtr* et *SSLConfigOffset* uniquement lorsque l'application émettant l'appel MQCONN s'exécute en tant que IBM MQ MQI client et que le protocole de canal est TCP/IP. Si

l'application n'est pas exécutée en tant que client IBM MQ ou que le protocole de canal n'est pas TCP/IP, *SSLConfigPtr* et *SSLConfigOffset* sont ignorés.

En spécifiant *SSLConfigPtr* ou *SSLConfigOffset*, plus *ClientConnPtr* ou *ClientConnOffset*, l'application peut contrôler l'utilisation de TLS pour la connexion client. Lorsque les informations TLS sont spécifiées de cette manière, les variables d'environnement MQSSLKEYR et MQSSLCRYP sont ignorées ; toutes les informations liées à TLS dans la table de définition de canal du client (CCDT) sont également ignorées.

Les informations TLS ne peuvent être spécifiées que sur:

- Premier appel MQCONNX du processus client, ou
- Un appel MQCONNX ultérieur lorsque toutes les connexions TLS précédentes au gestionnaire de files d'attente ont été conclues à l'aide de MQDISC.

Il s'agit des seuls états dans lesquels l'environnement TLS à l'échelle du processus peut être initialisé. Si un appel MQCONNX est émis en spécifiant des informations TLS alors que l'environnement TLS existe déjà, les informations TLS sur l'appel sont ignorées et la connexion est établie à l'aide de l'environnement TLS existant ; l'appel renvoie le code achèvement MQCC\_WARNING et le code anomalie MQRC\_SSL\_ALREADY\_INITIALISEE dans ce cas.

Vous pouvez fournir la structure MQSCO de la même manière que la structure MQCD, soit en spécifiant une adresse dans *SSLConfigPtr*, soit en spécifiant un décalage dans *SSLConfigOffset* ; voir la description de *ClientConnPtr* pour plus de détails sur la procédure à suivre. Toutefois, vous ne pouvez pas utiliser plus d'un *SSLConfigPtr* ou *SSLConfigOffset* ; l'appel échoue avec le code anomalie MQRC\_SSL\_CONFIG\_ERROR. si les deux sont différents de zéro.

Une fois l'appel MQCONNX terminé, la structure MQSCO n'est plus référencée.

Cette zone est ignorée si *Version* est inférieur à MQCNO\_VERSION\_4.

**Remarque :** Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

### ***SSLConfigOffset (MQLONG) pour MQCNO***

*SSLConfigOffset* est le décalage en octets d'une structure MQSCO depuis le début de la structure MQCNO. Le décalage peut être positif ou négatif. Cette zone est une zone d'entrée, avec une valeur initiale de 0.

Utilisez *SSLConfigOffset* uniquement lorsque l'application émettrice de l'appel MQCONNX s'exécute en tant que IBM MQ MQI client. Pour plus d'informations sur l'utilisation de cette zone, voir la description de la zone *SSLConfigPtr*.

Cette zone est ignorée si *Version* est inférieur à MQCNO\_VERSION\_4.

### ***ConnectionId (MQBYTE24) pour MQCNO***

*ConnectionId* est un identificateur unique de 24 octets qui permet à IBM MQ d'identifier de manière fiable une application. Une application peut utiliser cet identificateur pour la corrélation dans les appels PUT et GET. Ce paramètre de sortie a une valeur initiale de 24 octets nuls dans tous les langages de programmation.

Le gestionnaire de files d'attente affecte un ID unique à toutes les connexions, mais celles-ci sont établies. Si un MQCONNX établit la connexion avec un MQCNO de version 5, l'application peut déterminer l' *ConnectionId* à partir du MQCNO renvoyé. L'identificateur affecté est unique parmi tous les autres identificateurs générés par IBM MQ , tels que *CorrelId*, *MsgIDet GroupId*.

Utilisez *ConnectionId* pour identifier les unités d'oeuvre à exécution longue à l'aide de la commande PCF Inquire Connection ou de la commande MQSC DISPLAY CONN. L' *ConnectionId* utilisé par les commandes MQSC (CONN) est dérivé de l' *ConnectionId* renvoyé ici. Les commandes d'interrogation et d'arrêt de connexion PCF peuvent utiliser le *ConnectionId* renvoyé ici sans modification.

Vous pouvez utiliser l' `ConnectionId` pour forcer la fin d'une unité d'oeuvre de longue durée, en spécifiant l' `ConnectionId` à l'aide de la commande PCF Arrêter la connexion ou de la commande MQSC STOP CONN. Pour plus d'informations sur l'utilisation de ces commandes, voir [Arrêter la connexion](#) et [STOP CONN](#) .

Cette zone n'est pas renvoyée si la version est inférieure à `MQCNO_VERSION_5`.

La longueur de cette zone est indiquée par `MQ_CONNECTION_ID_LENGTH`.

### ***SecurityParmsDécalage (MQLONG) pour MQCNO***

`SecurityParmsLe décalage` est le décalage en octets de la structure MQCSP à partir du début de la structure MQCNO. Le décalage peut être positif ou négatif. Cette zone est une zone d'entrée, avec une valeur initiale de 0.

Cette zone est ignorée si `Version` est inférieure à `MQCNO_VERSION_5`.

La structure MQCSP est définie dans «MQCSP-Paramètres de sécurité», à la page 346.

### ***SecurityParmsPtr (PMQCSP) pour MQCNO***

`SecurityParmsPtr` est l'adresse de la structure MQCSP, utilisée pour spécifier un ID utilisateur et un mot de passe pour l'authentification par le service d'autorisation. Cette zone est une zone d'entrée et sa valeur initiale est un pointeur nul ou des octets nuls.

Cette zone est ignorée si `Version` est inférieure à `MQCNO_VERSION_5`.

La structure MQCSP est définie dans «MQCSP-Paramètres de sécurité», à la page 346.

### ***Réservé (MQBYTE4) pour MQCNO***

Zone réservée pour remplir la structure jusqu'à une limite de 64 bits. La valeur initiale de la zone est zéro binaire pour la longueur de la zone.

Cette zone est ignorée si `Version` est inférieur à `MQCNO_VERSION_6`.

### ***CCDTUrlLength (MQLONG) pour MQCNO***

`CCDTUrlLength` est la longueur de la chaîne identifiée par `CCDTUrlPtr` ou `CCDTUrlOffset` qui contient une URL qui identifie l'emplacement de la table de canaux de connexion client à utiliser pour la connexion. La valeur initiale de la zone est zéro.

Utilisez `CCDTUrlLength` uniquement lorsque l'application émettrice de l'appel MQCONNX s'exécute en tant que IBM MQ MQI client.

Il s'agit d'une alternative par programmation à la définition des variables d'environnement [MQCHLLIB](#) et [MQCHLTAB](#) .

Si l'application n'est pas exécutée en tant que client, `CCDTUrlLength` est ignoré.

Cette zone est ignorée si `Version` est inférieur à `MQCNO_VERSION_6`.

### ***CCDTUrlPtr (PMQCHAR) pour MQCNO***

`CCDTUrlPtr` est un pointeur facultatif vers une chaîne qui contient une URL, afin d'identifier l'emplacement de la table de canaux de connexion client à utiliser pour la connexion. Cette zone est une zone d'entrée, avec une valeur initiale d'un pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets all-null dans le cas contraire.

Utilisez `CCDTUrlPtr` uniquement lorsque l'application émettrice de l'appel MQCONNX s'exécute en tant que IBM MQ MQI client.

**Important :** Vous ne pouvez utiliser qu'un seul des éléments `CCDTUrlPtr` et `CCDTUrlOffset`. L'appel échoue avec le code anomalie `MQRC_CCDT_URL_ERROR` si les deux zones sont différentes de zéro.

Il s'agit d'une alternative par programmation à la définition des variables d'environnement [MQCHLLIB](#) et [MQCHLTAB](#) .

Si l'application n'est pas exécutée en tant que client, `CCDTUrlPtr` est ignoré.

Cette zone est ignorée si `Version` est inférieur à `MQCNO_VERSION_6`.



### **CCDTUrlOffset (MQLONG) pour MQCNO**

CCDTUrlOffset est le décalage en octets entre le début de la structure MQCNO et une chaîne contenant une URL qui identifie l'emplacement de la table de canaux de connexion client à utiliser pour la connexion. Le décalage peut être positif ou négatif et la valeur initiale de la zone est zéro.

Utilisez CCDTUrlOffset uniquement lorsque l'application émettrice de l'appel MQCONNX s'exécute en tant que IBM MQ MQI client.

**Important :** Vous ne pouvez utiliser qu'un seul des éléments CCDTUrlPtr et CCDTUrlOffset. L'appel échoue avec le code anomalie MQRC\_CCDT\_URL\_ERROR si les deux zones sont différentes de zéro.

Il s'agit d'une alternative par programmation à la définition des variables d'environnement [MQCHLLIB](#) et [MQCHLTAB](#).


Si l'application n'est pas exécutée en tant que client, CCDTUrlOffset est ignoré.

Cette zone est ignorée si Version est inférieur à MQCNO\_VERSION\_6.

### **AppName (MQCHAR28) pour MQCNO**

Nom défini par l'application pour identifier la connexion au gestionnaire de files d'attente. La valeur initiale de la zone est MQAN\_NONE\_ARRAY (caractères vides).

Cette zone est ignorée si Version est inférieur à MQCNO\_VERSION\_7 ou si la valeur est mise à blanc.

 Vous ne pouvez pas définir cette zone sur z/OS. Si vous tentez de le faire, vous recevez le code anomalie MQRC\_CNO\_ERROR en retour.

### **Reserved2 (MQBYTE4) pour MQCNO**

Zone réservée pour remplir la structure jusqu'à une limite de 64 bits. La valeur initiale de la zone est zéro binaire pour la longueur de la zone.

Cette zone est ignorée si Version est inférieur à MQCNO\_VERSION\_7.

### **BalanceParmsDécalage (MQLONG) pour MQCNO**

Emplacement de mémoire d'une structure de type MQBNO qui contient des informations sur le comportement d'équilibrage de l'application. La structure est entièrement ignorée, sauf si l'application se connecte via un canal client.

Cette zone est ignorée si Version est inférieur à MQCNO\_VERSION\_8.

Pour plus d'informations, voir [MQBNO](#).

Si vous renseignez cette zone, vous ne pouvez pas indiquer la zone «BalanceParmsPtr (MQPTR) pour MQCNO», à la page 345. Si vous tentez de fournir les deux zones, vous recevez une erreur MQRC\_CNO\_ERROR. Comme cette zone est pertinente uniquement pour les connexions client, la saisie de cette zone sur tout autre type de connexion génère également une erreur MQRC\_CNO\_ERROR.

### **BalanceParmsPtr (MQPTR) pour MQCNO**

Pointeur vers l'emplacement mémoire d'une structure de type MQBNO qui contient des informations sur le comportement d'équilibrage de l'application. La structure est entièrement ignorée, sauf si l'application se connecte via un canal client.

Cette zone est ignorée si Version est inférieur à MQCNO\_VERSION\_8.

Pour plus d'informations, voir [MQBNO](#).

Si vous renseignez cette zone, vous ne pouvez pas indiquer la zone «BalanceParmsDécalage (MQLONG) pour MQCNO», à la page 345. Si vous tentez de fournir les deux zones, vous recevez une erreur MQRC\_CNO\_ERROR. Comme cette zone est pertinente uniquement pour les connexions client, la saisie de cette zone sur tout autre type de connexion génère également une erreur MQRC\_CNO\_ERROR.

## MQCSP-Paramètres de sécurité

La structure des paramètres de sécurité de la connexion IBM MQ est utilisée par les applications pour transmettre des informations d'authentification sur un appel MQCONN au gestionnaire de files d'attente. Vous pouvez également l'utiliser pour fournir la clé initiale utilisée avec le système de protection par mot de passe IBM MQ qui chiffre les données sensibles.

Définissez *AuthenticationType* sur MQCSP\_AUTH\_USER\_ID\_AND\_PWD pour inclure l'ID utilisateur et le mot de passe de la version 1.

Lorsque vous fournissez des informations de clé initiales à la version 2, *AuthenticationType* prend par défaut la valeur MQCSP\_AUTH\_NONE.

**V 9.3.4** Depuis IBM MQ 9.3.4, utilisez *AuthenticationType* pour inclure des informations de jeton d'authentification.

**V 9.3.4** Vous pouvez utiliser MQCSP\_AUTH\_USER\_ID\_AND\_PWD ou MQCSP\_AUTH\_ID\_TOKEN, mais pas les deux.

**Avertissement :** Dans certains cas, le mot de passe ou le jeton d'authentification dans une structure MQCSP pour une application client est envoyé sur le réseau en texte en clair. Pour vous assurer que les mots de passe d'application client et les jetons d'authentification sont protégés de manière appropriée, voir [Protection par mot de passe MQCSP](#).

## Disponibilité

La structure MQCSP est disponible sur toutes les plateformes IBM MQ prises en charge.

## Version

Les fichiers d'en-tête, COPY et INCLUDE fournis pour les langages de programmation pris en charge contiennent la version la plus récente de MQCSP, mais avec la valeur initiale de la zone *Version* définie sur MQCSP\_VERSION\_1. Pour utiliser des zones qui ne sont pas présentes dans la structure version-1, l'application doit définir la zone *Version* sur le numéro de version requis.

## Jeu de caractères et codage

Les données dans MQCSP doivent être dans le jeu de caractères et le codage du gestionnaire de files d'attente local, qui sont fournis respectivement par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et par MQENC\_NATIVE.

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	MQCSP_STRUC_ID	'CSP'
<u>Version</u> (numéro de version de la structure)	MQCSP_VERSION_1	1
<u>AuthenticationType</u> (type d'authentification)	Aucun	MQCSP_AUTH_NONE
<u>Reserved1</u> (requis pour l'alignement des pointeurs sur IBM i)	Aucun	Chaîne nulle ou blancs

Tableau 482. Zones de MQCSP pour MQCSP (suite)









Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>CSPUserIdPtr</u> (adresse de l'ID utilisateur)	Aucun	Pointeur null ou octets null
<u>CSPUserIdDcalage</u> (décalage de l'ID utilisateur)	Aucun	0
<u>CSPUserIdLongueur</u> (longueur de l'ID utilisateur)	Aucun	0
<u>Reserved2</u> (obligatoire pour l'alignement des pointeurs sur IBM i)	Aucun	Chaîne nulle ou blancs
<u>CSPPasswordPtr</u> (adresse du mot de passe)	Aucun	Pointeur null ou octets null
<u>CSPPasswordOffset</u> (décalage du mot de passe)	Aucun	0
<u>CSPPasswordLength</u> (longueur du mot de passe)	Aucun	0
<b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieur à MQCSP_VERSION_2.		
 <u>Reserved3</u> (requis pour l'alignement des pointeurs sur IBM i)	Aucun	Chaîne nulle ou blancs
 <u>InitialKeyPtr</u>	Aucun	Pointeur null ou blancs
 <u>InitialKeyDcalage</u> (décalage de la clé initiale pour le système de protection par mot de passe)	Aucun	0
 <u>InitialKeyLongueur</u> (longueur de la clé initiale pour le système de protection par mot de passe)	Aucun	0
<b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieur à MQCSP_VERSION_3.		
 <u>Reserved4</u> (obligatoire pour l'alignement des pointeurs sur IBM i)	Aucun	Chaîne nulle ou blancs
 <u>TokenPtr</u> (adresse du jeton d'authentification)	Aucun	Pointeur null ou blancs
 <u>TokenKeyDcalage</u> (Décalage du jeton d'authentification)	Aucun	0
 <u>TokenLength</u> (longueur du jeton d'authentification)	Aucun	0

Tableau 482. Zones de MQCSP pour MQCSP (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<b>Remarques :</b>		
<p>1. Le symbole <code>\n</code> représente un caractère blanc unique.</p> <p>2. Dans le langage de programmation C, la variable macro <code>MQCSP_DEFAULT</code> contient les valeurs répertoriées dans la table. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</p>		
<pre>MQCSP MyCSP = {MQCSP_DEFAULT};</pre>		

## Déclarations de langue

Déclaration C pour MQCSP

```

V9.3.0 V9.3.0
typedef struct tagMQCSP MQCSP;
struct tagMQCSP {
    MQCHAR4    StructId;           /* Structure identifier */
    MQLONG     Version;           /* Structure version number */
    MQLONG     AuthenticationType; /* Type of authentication */
    MQBYTE4    Reserved1;        /* Required for IBM i pointer alignment */
    MQPTR      CSPUserIdPtr;      /* Address of user ID */
    MQLONG     CSPUserIdOffset;   /* Offset of user ID */
    MQLONG     CSPUserIdLength;   /* Length of user ID */
    MQBYTE8    Reserved2;        /* Required for IBM i pointer alignment */
    MQPTR      CSPPasswordPtr;    /* Address of password */
    MQLONG     CSPPasswordOffset; /* Offset of password */
    MQLONG     CSPPasswordLength; /* Length of password */
/* Ver:1 */

    MQBYTE8    Reserved3;        /* Required for IBM i pointer alignment */
    MQPTR      InitialKeyPtr;     /* Address of initial key */
    MQLONG     InitialKeyOffset;  /* Offset of initial key */
    MQLONG     InitialKeyLength;  /* Length of initial key */
/* Ver:2 */
};

```

```

V9.3.4
typedef struct tagMQCSP MQCSP;
struct tagMQCSP {
    MQCHAR4    StructId;           /* Structure identifier */
    MQLONG     Version;           /* Structure version number */
    MQLONG     AuthenticationType; /* Type of authentication */
    MQBYTE4    Reserved1;        /* Required for IBM i pointer alignment */
    MQPTR      CSPUserIdPtr;      /* Address of user ID */
    MQLONG     CSPUserIdOffset;   /* Offset of user ID */
    MQLONG     CSPUserIdLength;   /* Length of user ID */
    MQBYTE8    Reserved2;        /* Required for IBM i pointer alignment */
    MQPTR      CSPPasswordPtr;    /* Address of password */
    MQLONG     CSPPasswordOffset; /* Offset of password */
    MQLONG     CSPPasswordLength; /* Length of password */
/* Ver:1 */

    MQBYTE8    Reserved3;        /* Required for IBM i pointer alignment */
    MQPTR      InitialKeyPtr;     /* Address of initial key */
    MQLONG     InitialKeyOffset;  /* Offset of initial key */
    MQLONG     InitialKeyLength;  /* Length of initial key */
/* Ver:2 */

    MQBYTE8    Reserved4;        /* Required for IBM i pointer alignment */
    MQPTR      TokenPtr;          /* Address of token */
    MQLONG     TokenOffset;       /* Offset of token */
    MQLONG     TokenLength;       /* Length of token */

```

```
/* Ver:3 */  
};
```

## Déclaration COBOL pour MQCSP

```
V 9.3.0 V 9.3.0  
** MQCSP structure  
10 MQCSP.  
** Structure identifier  
15 MQCSP-STRUCID PIC X(4).  
** Structure version number  
15 MQCSP-VERSION PIC S9(9) BINARY.  
** Type of authentication  
15 MQCSP-AUTHENTICATIONTYPE PIC S9(9) BINARY.  
** Required for IBM i pointer alignment  
15 MQCSP-RESERVED1 PIC X(4).  
** Address of user ID  
15 MQCSP-CSPUSERIDPTR POINTER.  
** Offset of user ID  
15 MQCSP-CSPUSERIDOFFSET PIC S9(9) BINARY.  
** Length of user ID  
15 MQCSP-CSPUSERIDLENGTH PIC S9(9) BINARY.  
** Required for IBM i pointer alignment  
15 MQCSP-RESERVED2 PIC X(4).  
** Address of password  
15 MQCSP-CSPPASSWORDPTR POINTER.  
** Offset of password  
15 MQCSP-CSPPASSWORDOFFSET PIC S9(9) BINARY.  
** Length of password  
15 MQCSP-CSPPASSWORDLENGTH PIC S9(9) BINARY.  
** Ver:1 **  
  
** Reserved  
15 MQCSP-RESERVED3 PIC X(8).  
** Address of initial key  
15 MQCSP-INITIALKEYPTR POINTER.  
** Offset of initial key  
15 MQCSP-INITIALKEYOFFSET PIC S9(9) BINARY.  
** Length of initial key  
15 MQCSP-INITIALKEYLENGTH PIC S9(9) BINARY.  
** Ver:2 **
```

```
V 9.3.4  
** MQCSP structure  
10 MQCSP.  
** Structure identifier  
15 MQCSP-STRUCID PIC X(4).  
** Structure version number  
15 MQCSP-VERSION PIC S9(9) BINARY.  
** Type of authentication  
15 MQCSP-AUTHENTICATIONTYPE PIC S9(9) BINARY.  
** Required for IBM i pointer alignment  
15 MQCSP-RESERVED1 PIC X(4).  
** Address of user ID  
15 MQCSP-CSPUSERIDPTR POINTER.  
** Offset of user ID  
15 MQCSP-CSPUSERIDOFFSET PIC S9(9) BINARY.  
** Length of user ID  
15 MQCSP-CSPUSERIDLENGTH PIC S9(9) BINARY.  
** Required for IBM i pointer alignment  
15 MQCSP-RESERVED2 PIC X(4).  
** Address of password  
15 MQCSP-CSPPASSWORDPTR POINTER.  
** Offset of password  
15 MQCSP-CSPPASSWORDOFFSET PIC S9(9) BINARY.  
** Length of password  
15 MQCSP-CSPPASSWORDLENGTH PIC S9(9) BINARY.  
** Ver:1 **  
  
** Reserved  
15 MQCSP-RESERVED3 PIC X(8).  
** Address of initial key  
15 MQCSP-INITIALKEYPTR POINTER.  
** Offset of initial key  
15 MQCSP-INITIALKEYOFFSET PIC S9(9) BINARY.  
** Length of initial key  
15 MQCSP-INITIALKEYLENGTH PIC S9(9) BINARY.  
** Ver:2 **
```

```

**   Reserved
15 MQCSP-RESERVED4      PIC X(8).
**   Address of token
15 MQCSP-TOKENPTR      POINTER.
**   Offset of token
15 MQCSP-TOKENOFFSET   PIC S9(9) BINARY.
**   Length of token
15 MQCSP-TOKENLENGTH   PIC S9(9) BINARY.
** Ver:3 **

```

## Déclaration PL/I pour MQCSP

```

V 9.3.0 V 9.3.0
dcl
1 MQCSP based,
3 StrucId          char(4),          /* Structure identifier */
3 Version          fixed bin(31),    /* Structure version number */
3 AuthenticationType fixed bin(31),  /* Type of authentication */
3 Reserved1        char(4),          /* Required for IBM i pointer
                                     alignment */
3 CSPUserIdPtr     pointer,          /* Address of user ID */
3 CSPUserIdOffset  fixed bin(31),    /* Offset of user ID */
3 CSPUserIdLength  fixed bin(31),    /* Length of user ID */
3 Reserved2        char(8),          /* Required for IBM i pointer
                                     alignment */
3 CSPPasswordPtr   pointer,          /* Address of password */
3 CSPPasswordOffset fixed bin(31),   /* Offset of user ID */
3 CSPPasswordLength fixed bin(31),   /* Length of user ID */
/* Version 1 */

3 Reserved3        char(8),          /* Reserved */
3 InitialKeyPtr    pointer,          /* Address of initial key */
3 InitialKeyOffset fixed bin(31),    /* Offset of initial key */
3 InitialKeyLength fixed bin(31);    /* Length of initial key */
/* Version 2 */

```

```

V 9.3.4
dcl
1 MQCSP based,
3 StrucId          char(4),          /* Structure identifier */
3 Version          fixed bin(31),    /* Structure version number */
3 AuthenticationType fixed bin(31),  /* Type of authentication */
3 Reserved1        char(4),          /* Required for IBM i pointer
                                     alignment */
3 CSPUserIdPtr     pointer,          /* Address of user ID */
3 CSPUserIdOffset  fixed bin(31),    /* Offset of user ID */
3 CSPUserIdLength  fixed bin(31),    /* Length of user ID */
3 Reserved2        char(8),          /* Required for IBM i pointer
                                     alignment */
3 CSPPasswordPtr   pointer,          /* Address of password */
3 CSPPasswordOffset fixed bin(31),   /* Offset of user ID */
3 CSPPasswordLength fixed bin(31),   /* Length of user ID */
/* Version 1 */

3 Reserved3        char(8),          /* Reserved */
3 InitialKeyPtr    pointer,          /* Address of initial key */
3 InitialKeyOffset fixed bin(31),    /* Offset of initial key */
3 InitialKeyLength fixed bin(31);    /* Length of initial key */
/* Version 2 */

3 Reserved4        char(8),          /* Reserved */
3 TokenPtr         pointer,          /* Address of Token */
3 TokenOffset      fixed bin(31),    /* Offset of Token */
3 TokenLength      fixed bin(31);    /* Length of Token */
/* Version 3 */

```

## Déclaration Visual Basic pour MQCSP

```

Type MQCSP
StrucId          As String*4  'Structure identifier'
Version          As Long      'Structure version number'
AuthenticationType As Long    'Type of authentication'
Reserved1        As MQBYTE4   'Required for IBM i pointer
                               alignment'
CSPUserIdPtr     As MQPTR     'Address of user ID'

```

CSPUserIdOffset	As Long	'Offset of user ID'
CSPUserIdLength	As Long	'Length of user ID'
Reserved2	As MQBYTE8	'Required for IBM i pointer alignment'
CSPPasswordPtr	As MQPTR	'Address of password'
CSPPasswordOffset	As Long	'Offset of password'
CSPPasswordLength	As Long	'Length of password'
End Type		

## Concepts associés

[Utilisation des jetons d'authentification](#)

### **StrucId (MQCHAR4) pour MQCSP**

Il s'agit de l'identificateur de structure de la structure des paramètres de sécurité. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQCSP\_STRUC\_ID.

La valeur doit être:

#### **MQCSP\_STRUC\_ID**

Identificateur de la structure des paramètres de sécurité.

Pour le langage de programmation C, la constante MQCSP\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQCSP\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQCSP**

Numéro de version de la structure MQCSP.

La valeur doit être:

#### **MQCSP\_VERSION\_1**

Structure des paramètres de sécurité Version-1 . A la version 1, vous pouvez inclure un ID utilisateur et un mot de passe dans la structure MQCSP pour l'authentification auprès du gestionnaire de files d'attente.

#### **MQCSP\_VERSION\_2**

Structure des paramètres de sécurité Version-2 . A la version 2, vous pouvez inclure un ID utilisateur et un mot de passe pour vous authentifier auprès du gestionnaire de files d'attente et spécifier la clé initiale utilisée pour protéger les mots de passe.

#### **MQCSP\_VERSION\_3**

Structure des paramètres de sécurité Version-3 . A la version 3, vous pouvez inclure un ID utilisateur et un mot de passe ou un jeton d'authentification dans la structure MQCSP pour vous authentifier auprès du gestionnaire de files d'attente. Vous pouvez également spécifier la clé initiale utilisée pour protéger les mots de passe.

La constante suivante indique le numéro de version de la version en cours:

#### **MQCSP\_CURRENT\_VERSION**

Version actuelle de la structure des paramètres de sécurité.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQCSP\_VERSION\_3.

### **AuthenticationType (MQLONG) pour MQCSP**

AuthenticationType est une zone d'entrée. Sa valeur initiale est MQCSP\_AUTH\_NONE.

Il s'agit du type d'authentification à effectuer. Les valeurs valides sont :

#### **MQCSP\_AUTH\_NONE**

N'utilisez pas les zones d'ID utilisateur et de mot de passe ou de jeton d'authentification .

#### **MQCSP\_AUTH\_USER\_ID\_AND\_PWD**

Authentification à l'aide de l'ID utilisateur et du mot de passe dans la structure MQCSP.

#### **MQCSP\_AUTH\_ID\_TOKEN**

Authentification à l'aide du jeton d'authentification dans la structure MQCSP.

La valeur par défaut est MQCSP\_AUTH\_NONE. Avec le paramètre par défaut, aucune protection par mot de passe n'est effectuée.

Si vous avez besoin d'une authentification, vous devez définir **MQCSP.AuthenticationType** vers MQCSP\_AUTH\_USER\_ID\_AND\_PWD ou MQCSP\_AUTH\_ID\_TOKEN.

Pour plus d'informations, voir [Protection par mot de passe MQCSP](#).

### Concepts associés

[Utilisation des jetons d'authentification](#)

### **Reserved1 (MQBYTE4) pour MQCSP**

Zone réservée, requise pour l'alignement des pointeurs sur IBM i.

La valeur initiale de cette zone est nulle.

### **CSPUserIdPtr (MQPTR) pour MQCSP**

Adresse de l'ID utilisateur à utiliser pour l'authentification.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire. Cette zone est ignorée si *Version* est inférieur à MQCNO\_VERSION\_5.

Cette zone peut contenir un ID utilisateur de système d'exploitation lorsqu'un **AUTHTYPE** de IDPWOS est nommé dans la zone [CONNAUTH](#) du gestionnaire de files d'attente.

Sous Windows , il peut s'agir d'un ID utilisateur de domaine complet.

Cette zone peut contenir un ID utilisateur LDAP lorsqu'un **AUTHTYPE** de IDPWLDAP est nommé dans la zone [CONNAUTH](#) du gestionnaire de files d'attente.

### **CSPUserIdDécalage (MQLONG) pour MQCSP**


Décalage en octets de l'ID utilisateur à utiliser pour l'authentification. Le décalage peut être positif ou négatif.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

### **CSPUserIdLongueur (MQLONG) pour MQCSP**

Longueur de l'ID utilisateur à utiliser pour l'authentification.

La longueur maximale de l'ID utilisateur dépend de la plateforme. Voir [ID utilisateur](#).

 Si la longueur de l'ID utilisateur est supérieure à la longueur maximale autorisée, la demande d'authentification échoue avec MQRC\_CSP\_ERROR. Dans les versions antérieures de IBM MQ, l'erreur renvoyée est MQRC\_NOT\_AUTHORIZED.

Cette zone est une zone d'entrée. La valeur initiale de cette zone est 0.

### **Reserved2 (MQBYTE8) pour MQCSP**

Zone réservée, requise pour l'alignement des pointeurs sur IBM i.

La valeur initiale de cette zone est nulle.

### **CSPPasswordPtr (MQPTR) pour MQCSP**

Adresse du mot de passe à utiliser pour l'authentification.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire. Cette zone est ignorée si *Version* est inférieur à MQCNO\_VERSION\_5.



Cette zone peut contenir un mot de passe vide qui est rejeté par le système d'exploitation ou la vérification du mot de passe LDAP, selon la configuration, mais qui n'est pas rejeté par IBM MQ avant d'être transmis à la méthode d'authentification.

### **CSPPasswordOffset (MQLONG) pour MQCSP**



Décalage en octets du mot de passe à utiliser pour l'authentification. Le décalage peut être positif ou négatif.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

### **CSPPasswordLength (MQLONG) pour MQCSP**

Longueur du mot de passe à utiliser pour l'authentification.

La longueur maximale du mot de passe est MQ\_CSP\_PASSWORD\_LENGTH, soit 256 caractères.

  Si la longueur du mot de passe est supérieure à la longueur maximale autorisée, la demande d'authentification échoue avec MQRC\_CSP\_ERROR. Dans les versions antérieures de IBM MQ, l'erreur renvoyée est MQRC\_NOT\_AUTHORIZED.

Cette zone est une zone d'entrée. La valeur initiale de cette zone est 0.

### **Reserved3 (MQBYTE8) pour MQCSP**

Zone réservée, requise pour l'alignement du pointeur sur IBM i.

La valeur initiale de cette zone est nulle.

### **InitialKeyPtr (MQPTR) pour MQCSP**

Adresse de la clé initiale du système de protection par mot de passe.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire. Cette zone est ignorée si *Version* est inférieur à MQCSP\_VERSION\_2.

Cette zone est pertinente uniquement pour les IBM MQ MQI clients s'exécutant sur des systèmes IBM i, AIX, Linux, and Windows .

IBM MQ MQI clients peut fournir des valeurs chiffrées pour certaines zones, à l'aide du système de protection par mot de passe IBM MQ . Si vous avez utilisé une clé initiale pour chiffrer le mot de passe du référentiel de clés spécifié dans la structure MQCSO, veillez à inclure les zones de clé initiale dans le MQCSP pour la même application client.

Les clients IBM MQ MQI peuvent fournir des valeurs chiffrées pour certaines zones à l'aide du système de protection par mot de passe IBM MQ . Si vous avez utilisé une clé initiale pour chiffrer le mot de passe du référentiel de clés spécifié dans la structure MQCSO, veillez à inclure les zones de clé initiale dans la structure MQCSP pour la même application client.

Une clé initiale est utilisée par l'algorithme de chiffrement pour chiffrer et déchiffrer ces valeurs. Si une clé initiale est fournie lorsque les valeurs de ces zones sont chiffrées à l'aide de l'utilitaire **runmqicred** , la même clé initiale doit être spécifiée par le client lorsqu'il se connecte au gestionnaire de files d'attente.

La clé initiale spécifiée à l'aide de cette zone remplace toute clé initiale spécifiée à l'aide de la variable d'environnement *MQS\_MQI\_KEYFILE* ou de la propriété *MQIInitialKeyFile* dans la section Sécurité du fichier de configuration du client.

Vous pouvez utiliser *InitialKeyOffset* ou *InitialKeyPtr* pour spécifier la clé initiale, mais pas les deux.

#### **Tâches associées**

[Fourniture d'une clé initiale pour un client IBM MQ MQI sous AIX, Linux et Windows](#)

[Protection des mots de passe dans les fichiers de configuration du composant IBM MQ](#)

#### **Référence associée**

[runmqicred \(protection des mots de passe du client IBM MQ\)](#)

«KeyRepoPasswordPtr (MQPTR) pour MQSCO», à la page 593  
Il s'agit de l'adresse en octets de la phrase passe du référentiel de clés TLS.

«InitialKeyDécalage (MQLONG) pour MQCSP», à la page 354  
Décalage en octets de la clé initiale pour le système de protection par mot de passe à partir du début de la structure MQCSP. Le décalage peut être positif ou négatif.

**Multi** **V 9.3.0** **V 9.3.0** **InitialKeyDécalage (MQLONG) pour MQCSP**

Décalage en octets de la clé initiale pour le système de protection par mot de passe à partir du début de la structure MQCSP. Le décalage peut être positif ou négatif.

Vous pouvez utiliser *InitialKeyOffset* ou *InitialKeyPtr* pour spécifier la clé initiale, mais pas les deux. Pour plus d'informations, voir la description de la zone *InitialKeyPtr*.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQCSP\_VERSION\_2.

**Tâches associées**

[Protection des mots de passe dans les fichiers de configuration du composant IBM MQ](#)

[Fourniture d'une clé initiale pour un client IBM MQ MQI sous AIX, Linux et Windows](#)

**Référence associée**

«InitialKeyPtr (MQPTR) pour MQCSP», à la page 353

Adresse de la clé initiale du système de protection par mot de passe.

**Multi** **V 9.3.0** **V 9.3.0** **InitialKeyLongueur (MQLONG) pour MQCSP**

Longueur de la clé initiale pour le système de protection par mot de passe.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQCSP\_VERSION\_2.

**Tâches associées**

[Protection des mots de passe dans les fichiers de configuration du composant IBM MQ](#)

[Fourniture d'une clé initiale pour un client IBM MQ MQI sous AIX, Linux et Windows](#)

**V 9.3.4** **Reserved4 (MQBYTE8) pour MQCSP**

Zone réservée, requise pour l'alignement du pointeur sur IBM i.

La valeur initiale de cette zone est nulle.

**V 9.3.4** **TokenPtr (MQPTR) pour MQCSP**

Adresse du jeton d'authentification utilisé pour l'authentification auprès du gestionnaire de files d'attente.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement dans le cas contraire. Cette zone est ignorée si *Version* est inférieur à MQCSP\_VERSION\_3.

Cette zone est pertinente pour la IBM MQ MQI clients connexion aux gestionnaires de files d'attente IBM MQ qui s'exécutent sur des systèmes AIX ou Linux.

Vous pouvez utiliser *TokenOffset* ou *TokenPtr* pour spécifier le jeton d'authentification, mais pas les deux.

Pour plus d'informations, voir [Utilisation de jetons d'authentification dans une application](#).

**Concepts associés**

[Utilisation des jetons d'authentification](#)

**Référence associée**

«TokenOffset (MQLONG) pour MQCSP», à la page 355

Décalage en octets du jeton d'authentification à partir du début de la structure MQCSP. Le décalage peut être positif ou négatif.

### **V 9.3.4** *TokenOffset (MQLONG) pour MQCSP*

Décalage en octets du jeton d'authentification à partir du début de la structure MQCSP. Le décalage peut être positif ou négatif.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQCSP\_VERSION\_3.

Vous pouvez utiliser *TokenOffset* ou *TokenPtr* pour spécifier le jeton, mais pas les deux. Pour plus d'informations, voir la description de la zone *TokenPtr*.

#### **Concepts associés**

[Utilisation des jetons d'authentification](#)

#### **Tâches associées**

[Utilisation de jetons d'authentification dans une application](#)

#### **Référence associée**

«TokenPtr (MQPTR) pour MQCSP», à la page 354

Adresse du jeton d'authentification utilisé pour l'authentification auprès du gestionnaire de files d'attente.

### **V 9.3.4** *TokenLength (MQLONG) pour MQCSP*

Longueur du jeton d'authentification utilisé pour l'authentification auprès du gestionnaire de files d'attente.

La longueur maximale du jeton d'authentification est MQ\_CSP\_TOKEN\_LENGTH, qui est de 8192 octets. Si *TokenLength* est supérieur à la longueur maximale autorisée, la demande d'authentification échoue avec MQRC\_CSP\_ERROR.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQCSP\_VERSION\_3.

Pour plus d'informations, voir [Utilisation de jetons d'authentification dans une application](#).

#### **Concepts associés**

[Utilisation des jetons d'authentification](#)

## **MQCTLO-Structure des options de rappel de contrôle**

La structure MQCTLO permet de spécifier des options relatives à une fonction de rappel de contrôle. La structure est un paramètre d'entrée et de sortie dans l'appel MQCTL.

### **Disponibilité**

La structure MQCTLO est disponible sur les plateformes suivantes:

-  AIX
-  IBM i
-  Linux
-  Windows
-  z/OS

et pour IBM MQ MQI clients connecté à ces systèmes.

### **Version**

La version actuelle de MQCTLO est MQCTLO\_VERSION\_1.

## Jeu de caractères et codage

Les données de MQCTLO doivent se trouver dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par MQENC\_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
StrucID (identificateur de structure)	MQCTLO_STRUC_ID	'CTLO'
Version (numéro de version de la structure)	MQCTLO_VERSION_1	1
Options (options)	MQCTLO_NONE	Valeurs NULL
Options (zone réservée)	Réservé, zone	
ConnectionArea (zone à utiliser pour la fonction de rappel)	Aucun	Pointeur null ou octets null

**Remarques :**

- Dans le langage de programmation C, la variable macro MQCTLO\_DEFAULT contient les valeurs répertoriées dans la table. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQCTLO MyCTLO = {MQCTLO_DEFAULT};
```

## Déclarations de langue

Déclaration C pour MQCTLO

```
typedef struct tagMQCTLO MQCTLO;
struct tagMQCTLO {
    MQCHAR4  StrucId;          /* Structure identifier */
    MQLONG   Version;         /* Structure version number */
    MQLONG   Options;         /* Options that control the action of MQCTL */
    MQLONG   Reserved;        /* Reserved field */

    MQPTR    ConnectionArea; /* Connection work area passed to the function */
};
```

Déclaration COBOL pour MQCTLO

```
** MQCTLO structure
10 MQCTLO.
** Structure Identifier
15 MQCTLO-STRUCID                PIC X(4).
** Structure Version
15 MQCTLO-VERSION                PIC S9(9) BINARY.
** Options
15 MQCTLO-OPTIONS                PIC S9(9) BINARY.
** Reserved
15 MQCTLO-RESERVED                PIC S9(9) BINARY.
```

```

** ConnectionArea
15 MQCTLO-CONNECTIONAREA          POINTER

```

## Déclaration PL/I pour MQCTLO

```

dcl
1 MQCTLO based,
3 StructId          char(4),          /* Structure identifier */
3 Version           fixed bin(31),    /* Structure version */
3 Options           fixed bin(31),    /* Options */
3 Reserved          fixed bin(31),
3 ConnectionArea   pointer;          /* Connection work area */

```

### **StrucId (MQCHAR4) pour MQCTLO**

Il s'agit de l'identificateur de structure de la structure des options de contrôle. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQCTLO\_STRUC\_ID.

La valeur doit être:

#### **MQCTLO\_STRUC\_ID**

Identificateur de la structure des options de contrôle.

Pour le langage de programmation C, la constante MQCTLO\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQCTLO\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQCTLO**

Structure des options de contrôle-Zone Version

Il s'agit du numéro de version de la structure ; la valeur doit être:

#### **MQCTLO\_VERSION\_1**

Version-1 Structure des options de contrôle.

La constante suivante indique le numéro de version de la version en cours:

#### **MQCTLO\_CURRENT\_VERSION**

Version actuelle de la structure des options de contrôle.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQCTLO\_VERSION\_1.

### **Options (MQLONG) pour MQCTLO**

Structure des options de contrôle-Zone Options

Options qui contrôlent l'action de MQCTL.

#### **MQCTLO\_FAIL\_IF QUIESCING**

Forcez l'échec de l'appel MQCTL si le gestionnaire de files d'attente ou la connexion est à l'état de mise au repos.

Spécifiez MQGMO\_FAIL\_IF QUIESCING, dans les options MQGMO transmises à l'appel MQCB, pour envoyer une notification aux consommateurs de message lorsqu'ils sont mis au repos.

#### **MQCTLO\_AFFINITÉ UNITÉS d'exécution**

Cette option informe le système que l'application requiert que tous les consommateurs de messages, pour la même connexion, soient appelés sur la même unité d'exécution. Cette unité d'exécution sera utilisée pour tous les appels des consommateurs jusqu'à l'arrêt de la connexion.

**Option par défaut:** si vous n'avez besoin d'aucune des options décrites, utilisez l'option suivante:

#### **MQCTLO\_NONE**

Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. Toutes les options sont définies sur leur valeur par défaut. MQCTLO\_NONE est défini pour faciliter la documentation du programme ; il n'est pas prévu que cette option soit utilisée avec d'autres, mais comme sa valeur est zéro, une telle utilisation ne peut pas être détectée.

Il s'agit d'une zone d'entrée. La valeur initiale de la zone *Options* est MQCTLO\_NONE.

### **Réservé (MQLONG) pour MQCTLO**

Il s'agit d'une zone réservée. La valeur doit être zéro.

### **ConnectionArea (MQPTR) pour MQCTLO**

Structure des options de contrôle-Zone ConnectionArea

Il s'agit d'une zone disponible que la fonction de rappel peut utiliser.

Le gestionnaire de files d'attente ne prend aucune décision en fonction du contenu de cette zone et est transmis tel quel à la zone ConnectionArea de la structure MQCBC, qui est un paramètre d'entrée du rappel.

Cette zone est ignorée pour toutes les opérations autres que MQOP\_START et MQOP\_START\_WAIT.

Il s'agit d'une zone d'entrée et de sortie de la fonction de rappel. La valeur initiale de cette zone est un pointeur NULL ou des octets NULL.

## **En-tête MQDH-Distribution**

La structure MQDH décrit les données supplémentaires présentes dans un message lorsque ce message est un message de liste de distribution stocké dans une file d'attente de transmission. Un message de liste de distribution est un message envoyé à plusieurs files d'attente de destination. Les données supplémentaires sont constituées de la structure MQDH suivie d'un tableau d'enregistrements MQOR et d'un tableau d'enregistrements MQPMR. Cette structure est utilisée par les applications spécialisées qui placent les messages directement dans les files d'attente de transmission ou qui suppriment les messages des files d'attente de transmission (par exemple, les agents MCA). Les applications qui souhaitent placer des messages dans des listes de distribution ne doivent pas utiliser cette structure. A la place, ils doivent utiliser la structure MQOD pour définir les destinations dans la liste de distribution et la structure MQPMO pour spécifier les propriétés de message ou recevoir des informations sur les messages envoyés aux destinations individuelles.

## **Disponibilité**

La structure MQDH est disponible sur les plateformes suivantes:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour IBM MQ MQI clients connecté à ces systèmes.

## **Format**

MQFMT\_DIST\_HEADER

## **Jeu de caractères et codage**

Les données dans MQDH doivent se trouver dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par MQENC\_NATIVE.

Définissez le jeu de caractères et le codage de la MQDH dans les zones *CodedCharSetId* et *Encoding* dans:

- Le MQMD (si la structure MQDH est au début des données de message), ou

- Structure d'en-tête qui précède la structure MQDH (tous les autres cas).

## Utilisation

Lorsqu'une application insère un message dans une liste de distribution et que certaines ou toutes les destinations sont distantes, le gestionnaire de files d'attente préfixe les données de message d'application avec les structures MQXQH et MQDH et place le message dans la file d'attente de transmission appropriée. Les données se trouvent donc dans l'ordre suivant lorsque le message est dans une file d'attente de transmission:

- Structure MQXQH
- Structure MQDH plus tableaux d'enregistrements MQOR et MQPMR
- Données de message d'application

En fonction des destinations, le gestionnaire de files d'attente peut générer plusieurs messages de ce type et les placer dans des files d'attente de transmission différentes. Dans ce cas, les structures MQDH de ces messages identifient différents sous-ensembles des destinations définies par la liste de distribution ouverte par l'application.

Une application qui insère un message de liste de distribution directement dans une file d'attente de transmission doit se conformer à la séquence décrite précédemment et doit s'assurer que la structure MQDH est correcte. Si la structure MQDH n'est pas valide, le gestionnaire de files d'attente peut faire échouer l'appel MQPUT ou MQPUT1 avec le code anomalie MQRC\_DH\_ERROR.

Vous pouvez stocker des messages dans une file d'attente sous forme de liste de distribution uniquement si vous avez défini la file d'attente comme pouvant prendre en charge les messages de liste de distribution. Voir l'attribut de file d'attente **DistLists** décrit dans «Attributs des files d'attente», à la page 875. Si une application insère un message de liste de distribution directement dans une file d'attente qui ne prend pas en charge les listes de distribution, le gestionnaire de files d'attente divise le message de liste de distribution en messages individuels et place ces messages dans la file d'attente.

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID_STRUC_MQDH_	'DH--'
<u>Version</u> (numéro de version de la structure)	MQDH_VERSION_1	1
<u>StrucLength</u> (longueur de la structure MQDH plus les enregistrements suivants)	Aucun	0
<u>Codage</u> (codage numérique des données qui suit le tableau d'enregistrements MQPMR)	Aucun	0
<u>CodedCharSetId</u> (identificateur de jeu de caractères des données qui suivent le tableau des enregistrements MQPMR)	MQCCSI_UNDEFINI	0
<u>Format</u> (nom de format des données qui suivent un tableau d'enregistrements MQPMR)	MQFMT_AUCUN	Espaces vides
<u>Indicateurs</u> (indicateurs généraux)	MQDHF_NONE	0
<u>PutMsgRecFields</u> (indicateurs indiquant les zones MQPMR qui sont présentes)	MQPMRF_AUCUN	0

Tableau 484. Zones dans MQDH pour MQDH (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
RecsPresent (nombre d'enregistrements d'objet présents)	Aucun	0
ObjectRecOffset (décalage du premier enregistrement d'objet à partir du début de MQDH)	Aucun	0
PutMsgRecOffset (décalage du premier enregistrement d'insertion de message à partir du début de MQDH)	Aucun	0

**Remarques :**

1. Le symbole ~ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable macroMQDH\_DEFAULT contient les valeurs répertoriées dans la table. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQDH MyDH = {MQDH_DEFAULT};
```

## Déclarations de langue

### Déclaration C pour MQDH

```
typedef struct tagMQDH MQDH;
struct tagMQDH {
    MQCHAR4  StrucId;          /* Structure identifier */
    MQLONG   Version;        /* Structure version number */
    MQLONG   StrucLength;    /* Length of MQDH structure plus following
                             MQOR and MQPMPR records */
    MQLONG   Encoding;      /* Numeric encoding of data that follows
                             the MQOR and MQPMPR records */
    MQLONG   CodedCharSetId; /* Character set identifier of data that
                             follows the MQOR and MQPMPR records */
    MQCHAR8  Format;        /* Format name of data that follows the
                             MQOR and MQPMPR records */
    MQLONG   Flags;        /* General flags */
    MQLONG   PutMsgRecFields; /* Flags indicating which MQPMPR fields are
                             present */
    MQLONG   RecsPresent;   /* Number of MQOR records present */
    MQLONG   ObjectRecOffset; /* Offset of first MQOR record from start
                             of MQDH */
    MQLONG   PutMsgRecOffset; /* Offset of first MQPMPR record from start
                             of MQDH */
};
```

### Déclaration COBOL pour MQDH

```
** MQDH structure
10 MQDH.
** Structure identifier
15 MQDH-STRUCID PIC X(4).
** Structure version number
15 MQDH-VERSION PIC S9(9) BINARY.
** Length of MQDH structure plus following MQOR and MQPMPR records
15 MQDH-STRUCLength PIC S9(9) BINARY.
** Numeric encoding of data that follows the MQOR and MQPMPR records
15 MQDH-ENCODING PIC S9(9) BINARY.
** Character set identifier of data that follows the MQOR and MQPMPR
** records
15 MQDH-CODEDCHARSETID PIC S9(9) BINARY.
** Format name of data that follows the MQOR and MQPMPR records
```



```

15 MQDH-FORMAT          PIC X(8).
** General flags
15 MQDH-FLAGS          PIC S9(9) BINARY.
** Flags indicating which MQPMR fields are present
15 MQDH-PUTMSGRECFIELDS PIC S9(9) BINARY.
** Number of MQOR records present
15 MQDH-RECSPRESENT    PIC S9(9) BINARY.
** Offset of first MQOR record from start of MQDH
15 MQDH-OBJECTRECOFFSET PIC S9(9) BINARY.
** Offset of first MQPMR record from start of MQDH
15 MQDH-PUTMSGRECOFFSET PIC S9(9) BINARY.

```

## Déclaration PL/I pour MQDH

```

dcl
1 MQDH based,
3 StrucId          char(4),          /* Structure identifier */
3 Version          fixed bin(31), /* Structure version number */
3 StrucLength      fixed bin(31), /* Length of MQDH structure plus
                                following MQOR and MQPMR
                                records */
3 Encoding         fixed bin(31), /* Numeric encoding of data that
                                follows the MQOR and MQPMR
                                records */
3 CodedCharSetId  fixed bin(31), /* Character set identifier of data
                                that follows the MQOR and MQPMR
                                records */
3 Format           char(8),          /* Format name of data that follows
                                the MQOR and MQPMR records */
3 Flags           fixed bin(31), /* General flags */
3 PutMsgRecFields fixed bin(31), /* Flags indicating which MQPMR
                                fields are present */
3 RecsPresent     fixed bin(31), /* Number of MQOR records present */
3 ObjectRecOffset fixed bin(31), /* Offset of first MQOR record from
                                start of MQDH */
3 PutMsgRecOffset fixed bin(31); /* Offset of first MQPMR record from
                                start of MQDH */

```

## Déclaration Visual Basic pour MQDH

```

Type MQDH
StrucId          As String*4 'Structure identifier'
Version          As Long     'Structure version number'
StrucLength      As Long     'Length of MQDH structure plus following'
                                'MQOR and MQPMR records'
Encoding         As Long     'Numeric encoding of data that follows'
                                'the MQOR and MQPMR records'
CodedCharSetId  As Long     'Character set identifier of data that'
                                'follows the MQOR and MQPMR records'
Format           As String*8 'Format name of data that follows the'
                                'MQOR and MQPMR records'
Flags           As Long     'General flags'
PutMsgRecFields As Long     'Flags indicating which MQPMR fields are'
                                'present'
RecsPresent     As Long     'Number of MQOR records present'
ObjectRecOffset As Long     'Offset of first MQOR record from start'
                                'of MQDH'
PutMsgRecOffset As Long     'Offset of first MQPMR record from start'
                                'of MQDH'
End Type

```

### **StrucId (MQCHAR4) pour MQDH**

Il s'agit de l'identificateur de structure de la structure d'en-tête de distribution. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQDH\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUC\_MQDH\_**

Identificateur de la structure d'en-tête de distribution.

Pour le langage de programmation C, la constante MQDH\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQDH\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQDH**

La valeur doit être:

#### **MQDH\_VERSION\_1**

Numéro de version de la structure d'en-tête de distribution.

La constante suivante indique le numéro de version de la version en cours:

#### **MQDH\_CURRENT\_VERSION**

Version actuelle de la structure d'en-tête de distribution.

La valeur initiale de cette zone est MQDH\_VERSION\_1.

### **StrucLength (MQLONG) pour MQDH**

Il s'agit du nombre d'octets entre le début de la structure MQDH et le début des données de message suivant les tableaux d'enregistrements MQOR et MQPMR. Les données se produisent dans l'ordre suivant:

- Structure MQDH
- Tableau d'enregistrements MQOR
- Tableau d'enregistrements MQPMR
- Données de message

Les tableaux des enregistrements MQOR et MQPMR sont traités par des décalages contenus dans la structure MQDH. Si ces décalages génèrent des octets inutilisés entre une ou plusieurs structures MQDH, les tableaux d'enregistrements et les données de message, ces octets inutilisés doivent être inclus dans la valeur de *StrucLength*, mais le contenu de ces octets n'est pas conservé par le gestionnaire de files d'attente. Le tableau des enregistrements MQPMR doit précéder le tableau des enregistrements MQOR.

La valeur initiale de cette zone est 0.

### **Codage (MQLONG) pour MQDH**

Il s'agit du codage numérique des données qui suit les tableaux des enregistrements MQOR et MQPMR ; il ne s'applique pas aux données numériques de la structure MQDH elle-même.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données.

La valeur initiale de cette zone est 0.

### **CodedCharSetId (MQLONG) pour MQDH**

Il s'agit de l'identificateur de jeu de caractères des données qui suivent les tableaux des enregistrements MQOR et MQPMR ; il ne s'applique pas aux données de type caractère dans la structure MQDH elle-même.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données. Vous pouvez utiliser la valeur spéciale suivante:

#### **MQCCSI\_HÉRITER**

Hérite de l'identificateur de jeu de caractères de cette structure.

Les données de type caractères dans les données *suivant* cette structure se trouvent dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, l'appel MQGET ne renvoie pas la valeur MQCCSI\_INHERIT.

Vous ne pouvez pas utiliser MQCCSI\_INHERIT si la valeur de la zone *PutApplType* dans MQMD est MQAT\_BROKER.

Cette valeur est prise en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour les clients IBM MQ connectés à ces systèmes.

La valeur initiale de cette zone est MQCCSI\_UNDEFINED.

### **Format (MQCHAR8) pour MQDH**

Il s'agit du nom de format des données qui suivent les tableaux des enregistrements MQOD et MQPMR (selon la dernière occurrence).

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de la zone *Format* dans MQMD.

La valeur initiale de cette zone est MQFMT\_NONE.

### **Indicateurs (MQLONG) pour MQDH**

Vous pouvez spécifier l'indicateur suivant:

#### **MQDHF\_NEW\_MSG\_IDS**

Générez un nouvel identificateur de message pour chaque destination de la liste de distribution. Définissez cette option uniquement si aucun enregistrement de message d'insertion n'est présent ou si les enregistrements sont présents mais qu'ils ne contiennent pas la zone *MsgId*.

L'utilisation de cet indicateur diffère la génération des identificateurs de message jusqu'au moment où le message de liste de distribution est finalement divisé en messages individuels. Cela réduit la quantité d'informations de contrôle qui doivent être transmises avec le message de liste de distribution.

Lorsqu'une application insère un message dans une liste de distribution, le gestionnaire de files d'attente définit MQDHF\_NEW\_MSG\_IDS dans la MQDH qu'elle génère lorsque les deux instructions suivantes sont vraies:

- Aucun enregistrement d'insertion de message n'est fourni par l'application ou les enregistrements fournis ne contiennent pas la zone *MsgId*.
- La zone *MsgId* dans MQMD est MQMI\_NONE ou la zone *Options* dans MQPMO inclut MQPMO\_NEW\_MSG\_ID

Si aucun indicateur n'est nécessaire, spécifiez ce qui suit:

#### **MQDHF\_NONE**

Aucun indicateur n'a été spécifié. MQDHF\_NONE est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette constante soit utilisée avec d'autres, mais comme sa valeur est nulle, une telle utilisation ne peut pas être détectée.

La valeur initiale de cette zone est MQDHF\_NONE.

### **PutMsgRecFields (MQLONG) pour MQDH**

Vous pouvez spécifier aucun ou plusieurs des indicateurs suivants:

#### **ID MQPMRF\_MSG\_ID**

La zone d'identificateur de message est présente.

#### **ID\_CORREL\_MQPMRF\_**

La zone d'identificateur de corrélation est présente.

#### **ID\_GROUPE\_MQPMRF\_ID**

La zone d'identificateur de groupe est présente.

## **MQPMRF\_FEEDBACK**

Un champ de commentaires est présent.

## **MQPMRF\_COMPTING\_TOKEN**

La zone de jeton de comptabilité est présente.

Si aucune zone MQPMR n'est présente, indiquez ce qui suit:

## **MQPMRF\_AUCUN**

Aucune zone d'enregistrement d'insertion de message n'est présente. MQPMRF\_NONE est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette constante soit utilisée avec d'autres, mais comme sa valeur est nulle, une telle utilisation ne peut pas être détectée.

La valeur initiale de cette zone est MQPMRF\_NONE.

## ***RecsPresent (MQLONG) pour MQDH***

Il s'agit du nombre de destinations. Une liste de distribution doit toujours contenir au moins une destination. Par conséquent, *RecsPresent* doit toujours être supérieur à zéro.

La valeur initiale de cette zone est 0.

## ***ObjectRecDécalage (MQLONG) pour MQDH***

Indique le décalage en octets du premier enregistrement du tableau des enregistrements d'objet MQOR contenant les noms des files d'attente de destination. Ce tableau contient *RecsPresent* enregistrements. Ces enregistrements (plus les octets ignorés entre le premier enregistrement d'objet et la zone précédente) sont inclus dans la longueur indiquée par la zone *StrucLength*.

Une liste de distribution doit toujours contenir au moins une destination. Par conséquent, *ObjectRecOffset* doit toujours être supérieur à zéro.

La valeur initiale de cette zone est 0.

## ***PutMsgRecOffset (MQLONG) pour MQDH***

Indique le décalage en octets du premier enregistrement dans le tableau des enregistrements de message d'insertion MQPMR contenant les propriétés de message. S'il est présent, ce tableau contient *RecsPresent* enregistrements. Ces enregistrements (plus les octets ignorés entre le premier enregistrement de message inséré et la zone précédente) sont inclus dans la longueur indiquée par la zone *StrucLength*.

Les enregistrements de message d'insertion sont facultatifs ; si aucun enregistrement n'est fourni, *PutMsgRecOffset* est égal à zéro et *PutMsgRecFields* a la valeur MQPMRF\_NONE.

La valeur initiale de cette zone est 0.

## **MQDLH-En-tête de rebut**

La structure MQDLH décrit les informations qui préfixes les données de message d'application des messages de la file d'attente des messages non livrés. Un message peut arriver dans la file d'attente de rebut soit parce que le gestionnaire de files d'attente ou l'agent de canal de message l'a redirigé vers la file d'attente, soit parce qu'une application a inséré le message directement dans la file d'attente.

## **Format**

MQFMT\_DEAD\_LETTER\_HEADER

## **Jeu de caractères et codage**

Les zones de la structure MQDLH sont dans le jeu de caractères et le codage donnés par les zones *CodedCharSetId* et *Encoding*. Ils sont spécifiés dans la structure d'en-tête qui précède le MQDLH, ou dans la structure MQMD si le MQDLH se trouve au début des données de message d'application.

Le jeu de caractères doit comporter des caractères mono-octet pour les caractères admis dans les noms de file d'attente.

Si vous utilisez les classes IBM MQ pour Java/JMSet que la page de codes définie dans le MQMD n'est pas prise en charge par la machine virtuelle Java, le MQDLH est écrit dans le jeu de caractères UTF-8.

## Utilisation

Les applications qui placent des messages directement dans la file d'attente de messages non livrés doivent ajouter une structure MQDLH aux données de message et initialiser les zones avec les valeurs appropriées. Toutefois, le gestionnaire de files d'attente ne requiert pas la présence d'une structure MQDLH ou la spécification de valeurs valides pour les zones.

Si un message est trop long pour être inséré dans la file d'attente de rebut, l'application doit effectuer l'une des opérations suivantes:

- Tronquez les données de message pour qu'elles tiennent dans la file d'attente des messages non livrés.
- Enregistrez le message dans la mémoire secondaire et placez un message de rapport d'exception dans la file d'attente des messages non livrés.
- Supprimez le message et renvoyez une erreur à son émetteur. Si le message est (ou peut être) un message critique, ne le faites que s'il est connu que l'émetteur possède toujours une copie du message ; par exemple, un message reçu par un agent de canal de transmission à partir d'un canal de communication.

Parmi les actions précédentes, celle qui est appropriée (le cas échéant) dépend de la conception de l'application.

Le gestionnaire de files d'attente effectue un traitement spécial lorsqu'un message qui est un segment est inséré avec une structure MQDLH à l'avant. Pour plus de détails, voir la description de la structure MQMDE.

## Insertion de messages dans la file d'attente des messages non livrés

Lorsqu'un message est inséré dans la file d'attente de rebut, la structure MQMD utilisée pour l'appel MQPUT ou MQPUT1 doit être identique au MQMD associé au message (généralement le MQMD renvoyé par l'appel MQGET), à l'exception de ce qui suit:

- Définissez les zones *CodedCharSetId* et *Encoding* sur le jeu de caractères et le codage utilisés pour les zones de la structure MQDLH.
- Définissez la zone *Format* sur MQFMT\_DEAD\_LETTER\_HEADER pour indiquer que les données commencent par une structure MQDLH.
- Définissez les zones de contexte (*AccountingToken*, *AppIdentityData*, *AppOriginData*, *PutAppName*, *PutAppType*, *PutDate*, *PutTime*, *UserIdentifier*) à l'aide d'une option de contexte adaptée aux circonstances:
  - Une application qui insère dans la file d'attente de rebut un message qui n'est lié à aucun message précédent doit utiliser l'option MQPMO\_DEFAULT\_CONTEXT ; cela permet au gestionnaire de files d'attente de définir toutes les zones de contexte du descripteur de message sur leurs valeurs par défaut.
  - Une application serveur plaçant dans la file d'attente de rebut un message qu'elle vient de recevoir doit utiliser l'option MQPMO\_PASS\_ALL\_CONTEXT pour conserver les informations de contexte d'origine.
  - Une application serveur qui place dans la file d'attente des messages non livrés une *réponse* à un message qu'elle vient de recevoir doit utiliser l'option MQPMO\_PASS\_IDENTITY\_CONTEXT ; cette option conserve les informations d'identité mais définit les informations d'origine comme étant celles de l'application serveur.
  - Un agent de canal de transmission de messages plaçant dans la file d'attente de messages non livrés un message qu'il a reçu de son canal de communication doit utiliser l'option MQPMO\_SET\_ALL\_CONTEXT pour conserver les informations de contexte d'origine.

Dans la structure MQDLH elle-même, définissez les zones comme suit:

- Définissez les zones *CodedCharSetId*, *Encodinget Format* sur les valeurs qui décrivent les données qui suivent la structure MQDLH, généralement les valeurs du descripteur de message d'origine.
- Définissez les zones de contexte *PutApplType*, *PutApplName*, *PutDateet PutTime* sur les valeurs appropriées à l'application qui place le message dans la file d'attente de rebut ; ces valeurs ne sont pas liées au message d'origine.
- Définissez d'autres zones selon les besoins.

Vérifiez que toutes les zones ont des valeurs valides et que les zones alphanumériques sont remplies avec des blancs à la longueur définie de la zone ; ne mettez pas fin aux données alphanumériques prématurément en utilisant un caractère null, car le gestionnaire de files d'attente ne convertit pas les caractères null et suivants en blancs dans la structure MQDLH.

## Obtention de messages de la file d'attente des messages non livrés

Les applications qui extraient des messages de la file d'attente de rebut doivent vérifier que les messages commencent par une structure MQDLH. L'application peut déterminer si une structure MQDLH est présente en examinant la zone *Format* dans le descripteur de message MQMD ; si la zone a la valeur MQFMT\_DEAD\_LETTER\_HEADER, les données de message commencent par une structure MQDLH. Notez également que les messages que les applications reçoivent de la file d'attente de messages non livrés peuvent être tronqués s'ils étaient à l'origine trop longs pour la file d'attente.

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID_STRUC_MQDLH	'DLH~'
<u>Version</u> (numéro de version de la structure)	MQDLH_VERSION_1	1
<u>Motif</u> (le message du motif est arrivé dans la file d'attente des messages non livrés)	MQRC_NONE	0
<u>DestQName</u> (nom de la file d'attente de destination d'origine)	Aucun	Chaîne nulle ou blancs
<u>DestQMgr</u> (nom du gestionnaire de files d'attente de destination d'origine)	Aucun	Chaîne nulle ou blancs
<u>Codage</u> (codage numérique des données qui suit MQDLH)	Aucun	0
<u>CodedCharSetId</u> (identificateur de jeu de caractères des données qui suit MQDLH)	MQCCSI_UNDEFINI	0
<u>Format</u> (nom de format des données qui suivent MQDLH)	MQFMT_AUCUN	Espaces vides
<u>PutApplType</u> (type d'application qui place le message dans la file d'attente de rebut)	Aucun	0
<u>PutApplNom</u> (nom de l'application qui a inséré le message dans la file d'attente de rebut)	Aucun	Chaîne nulle ou blancs

Tableau 485. Zones de MQDLH pour MQDLH (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
PutDate (date à laquelle le message a été inséré dans la file d'attente de rebut)	Aucun	Chaîne nulle ou blancs
PutTime (heure à laquelle le message a été inséré dans la file d'attente de rebut)	Aucun	Chaîne nulle ou blancs
<p><b>Remarques :</b></p> <ol style="list-style-type: none"> <li>1. Le symbole ~ représente un caractère blanc unique.</li> <li>2. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.</li> <li>3. Dans le langage de programmation C, la variable macroMQDLH_DEFAULT contient les valeurs répertoriées dans la table. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</li> </ol> <pre>MQDLH MyDLH = {MQDLH_DEFAULT};</pre>		

## Déclarations de langue

### Déclaration C pour MQDLH

```
typedef struct tagMQDLH MQDLH;
struct tagMQDLH {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;         /* Structure version number */
    MQLONG    Reason;          /* Reason message arrived on dead-letter
                               (undelivered-message) queue */
    MQCHAR48  DestQName;       /* Name of original destination queue */
    MQCHAR48  DestQMgrName;    /* Name of original destination queue
                               manager */
    MQLONG    Encoding;        /* Numeric encoding of data that follows
                               MQDLH */
    MQLONG    CodedCharSetId; /* Character set identifier of data that
                               follows MQDLH */
    MQCHAR8   Format;          /* Format name of data that follows
                               MQDLH */
    MQLONG    PutApplType;     /* Type of application that put message on
                               dead-letter (undelivered-message)
                               queue */
    MQCHAR28  PutApplName;     /* Name of application that put message on
                               dead-letter (undelivered-message)
                               queue */
    MQCHAR8   PutDate;         /* Date when message was put on dead-letter
                               (undelivered-message) queue */
    MQCHAR8   PutTime;        /* Time when message was put on the
                               dead-letter (undelivered-message)
                               queue */
};
```

### Déclaration COBOL pour MQDLH

```
** MQDLH structure
   10 MQDLH.
**   Structure identifier
   15 MQDLH-STRUCID      PIC X(4).
**   Structure version number
   15 MQDLH-VERSION     PIC S9(9) BINARY.
**   Reason message arrived on dead-letter (undelivered-message) queue
   15 MQDLH-REASON     PIC S9(9) BINARY.
**   Name of original destination queue
   15 MQDLH-DESTQNAME   PIC X(48).
```

```

** Name of original destination queue manager
15 MQDLH-DESTQMgrNAME PIC X(48).
** Numeric encoding of data that follows MQDLH
15 MQDLH-ENCODING PIC S9(9) BINARY.
** Character set identifier of data that follows MQDLH
15 MQDLH-CODEDCHARSETID PIC S9(9) BINARY.
** Format name of data that follows MQDLH
15 MQDLH-FORMAT PIC X(8).
** Type of application that put message on dead-letter
** (undelivered-message) queue
15 MQDLH-PUTAPPLTYPE PIC S9(9) BINARY.
** Name of application that put message on dead-letter
** (undelivered-message) queue
15 MQDLH-PUTAPPLNAME PIC X(28).
** Date when message was put on dead-letter (undelivered-message)
** queue
15 MQDLH-PUTDATE PIC X(8).
** Time when message was put on the dead-letter (undelivered-message)
** queue
15 MQDLH-PUTTIME PIC X(8).

```

### Déclaration PL/I pour MQDLH

```

dcl
  1 MQDLH based,
  3 StrucId          char(4),          /* Structure identifier */
  3 Version          fixed bin(31), /* Structure version number */
  3 Reason          fixed bin(31), /* Reason message arrived on
                                dead-letter (undelivered-message)
                                queue */
  3 DestQName       char(48),          /* Name of original destination
                                queue */
  3 DestQMgrName    char(48),          /* Name of original destination queue
                                manager */
  3 Encoding        fixed bin(31), /* Numeric encoding of data that
                                follows MQDLH */
  3 CodedCharSetId fixed bin(31), /* Character set identifier of data
                                that follows MQDLH */
  3 Format           char(8),          /* Format name of data that follows
                                MQDLH */
  3 PutApplType     fixed bin(31), /* Type of application that put
                                message on dead-letter
                                (undelivered-message) queue */
  3 PutApplName     char(28),          /* Name of application that put
                                message on dead-letter
                                (undelivered-message) queue */
  3 PutDate         char(8),          /* Date when message was put on
                                dead-letter (undelivered-message)
                                queue */
  3 PutTime         char(8);          /* Time when message was put on the
                                dead-letter (undelivered-message)
                                queue */

```

### Déclaration High Level Assembler pour MQDLH

```

MQDLH          DSECT
MQDLH_STRUCID  DS CL4  Structure identifier
MQDLH_VERSION  DS F    Structure version number
MQDLH_REASON   DS F    Reason message arrived on dead-letter
*              (undelivered-message) queue
MQDLH_DESTQNAME DS CL48 Name of original destination queue
MQDLH_DESTQMgrNAME DS CL48 Name of original destination queue
*              manager
MQDLH_ENCODING DS F    Numeric encoding of data that follows
*              MQDLH
MQDLH_CODEDCHARSETID DS F Character set identifier of data that
*              follows MQDLH
MQDLH_FORMAT   DS CL8  Format name of data that follows MQDLH
MQDLH_PUTAPPLTYPE DS F  Type of application that put message on
*              dead-letter (undelivered-message) queue
MQDLH_PUTAPPLNAME DS CL28 Name of application that put message on
*              dead-letter (undelivered-message) queue
MQDLH_PUTDATE  DS CL8  Date when message was put on
*              dead-letter (undelivered-message) queue
MQDLH_PUTTIME  DS CL8  Time when message was put on the
*              dead-letter (undelivered-message) queue
*
*

```



MQDLH_LENGTH	EQU	*-MQDLH
	ORG	MQDLH
MQDLH_AREA	DS	CL(MQDLH_LENGTH)

## Déclaration Visual Basic pour MQDLH

```

Type MQDLH
  StrucId      As String*4  'Structure identifier'
  Version      As Long      'Structure version number'
  Reason       As Long      'Reason message arrived on dead-letter'
                '(undelivered-message) queue'
  DestQName    As String*48 'Name of original destination queue'
  DestQMgrName As String*48 'Name of original destination queue'
                'manager'
  Encoding     As Long      'Numeric encoding of data that follows'
                'MQDLH'
  CodedCharSetId As Long    'Character set identifier of data that'
                'follows MQDLH'
  Format       As String*8  'Format name of data that follows MQDLH'
  PutApplType  As Long      'Type of application that put message on'
                'dead-letter (undelivered-message) queue'
  PutApplName  As String*28 'Name of application that put message on'
                'dead-letter (undelivered-message) queue'
  PutDate      As String*8  'Date when message was put on dead-letter'
                '(undelivered-message) queue'
  PutTime      As String*8  'Time when message was put on the'
                'dead-letter (undelivered-message) queue'
End Type

```

### **StrucId (MQCHAR4) pour MQDLH**

Il s'agit de l'identificateur de structure de la structure d'en-tête de rebut. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQDLH\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUC\_MQDLH**

Identificateur de la structure d'en-tête de rebut.

Pour le langage de programmation C, la constante MQDLH\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQDLH\_STRUC\_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

### **Version (MQLONG) pour MQDLH**

La version est le numéro de version de la structure.

La valeur doit être:

#### **MQDLH\_VERSION\_1**

Numéro de version de la structure d'en-tête de rebut.

La constante suivante indique le numéro de version de la version en cours:

#### **MQDLH\_CURRENT\_VERSION**

Version actuelle de la structure d'en-tête de rebut.

La valeur initiale de cette zone est MQDLH\_VERSION\_1.

### **Motif (MQLONG) pour MQDLH**

La zone Raison identifie la raison pour laquelle le message a été placé dans la file d'attente de rebut et non dans la file d'attente de destination d'origine.

Identifie la raison pour laquelle le message a été placé dans la file d'attente de rebut au lieu de la file d'attente de destination d'origine. Il doit s'agir de l'une des valeurs MQFB\_\* ou MQRC\_\* (par exemple, MQRC\_Q\_FULL). Voir la description de la zone *Feedback* dans «MQMD-Descripteur de message», à la page 438 pour plus de détails sur les valeurs MQFB\_\* communes qui peuvent se produire.

Si la valeur est comprise entre MQFB\_IMS\_FIRST et MQFB\_IMS\_LAST, le code d'erreur IMS réel peut être déterminé en soustrayant MQFB\_IMS\_ERROR de la valeur de la zone *Reason*.

Certaines valeurs MQFB\_\* apparaissent uniquement dans cette zone. Ils concernent des messages de référentiel, des messages de déclenchement ou des messages de file d'attente de transmission qui ont été transférés dans la file d'attente de rebut. Il s'agit des fonctions suivantes :

**MQFB\_APPL\_CANNOT\_BE\_STARTED ( X'00000109 ')**

Une application traitant un message de déclenchement ne peut pas démarrer l'application nommée dans la zone *AppLId* du message de déclenchement (voir «MQTM-Message de déclenchement», à la page 627).

Sous z/OS, la transaction CKTI CICS est un exemple d'application qui traite les messages de déclenchement.

**MQFB\_APPL\_TYPE\_ERROR ( X'0000010B ')**

Une application traitant un message de déclenchement ne peut pas démarrer l'application car la zone *AppLType* du message de déclenchement n'est pas valide (voir «MQTM-Message de déclenchement», à la page 627).

Sous z/OS, la transaction CKTI CICS est un exemple d'application qui traite les messages de déclenchement.

**MQFB\_BIND\_OPEN\_CLUSRCVR\_DEL ( X'00000119 ')**

Le message se trouvait dans SYSTEM.CLUSTER.TRANSMIT.QUEUE destiné à une file d'attente de cluster qui a été ouverte avec l'option MQOO\_BIND\_ON\_OPEN, mais le canal récepteur de cluster distant à utiliser pour transmettre le message à la file d'attente de destination a été supprimé avant que le message ne puisse être envoyé. MQOO\_BIND\_ON\_OPEN ayant été indiqué, seul le canal sélectionné lors de l'ouverture de la file d'attente peut être utilisé pour transmettre le message. Ce canal n'étant plus disponible, le message est placé dans la file d'attente des messages non livrés.

**MQFB\_NOT\_A\_REPOSITORY\_MSG ( X'00000118 ')**

Le message n'est pas un message de référentiel.

**MQFB\_STOPPED\_BY\_CHAD\_EXIT ( X'00000115 ')**

Le message a été arrêté par l'exit de définition automatique de canal.

**MQFB\_STOPPED\_BY\_MSG\_EXIT ( X'0000010D ')**

Le message a été arrêté par l'exit de message de canal.

**MQFB\_TM\_ERROR ( X'0000010A ')**

La zone *Format* de MQMD indique MQFMT\_TRIGGER, mais le message ne commence pas par une structure MQTM valide. Par exemple, il se peut que l'identificateur mnémorique *StrucId* ne soit pas valide, que *Version* ne soit pas reconnu ou que la longueur du message de déclenchement ne soit pas suffisante pour contenir la structure MQTM.

Sous z/OS, la transaction CKTI CICS est un exemple d'application qui traite les messages de déclenchement et peut générer ce code retour.

**MQFB\_XMIT\_Q\_MSG\_ERROR ( X'0000010F ')**

Un agent MCA a détecté qu'un message de la file d'attente de transmission n'est pas au format correct. L'agent MCA place le message dans la file d'attente de rebut à l'aide de ce code retour.

Une cause courante est qu'un message a été inséré directement dans la file d'attente de transmission, de sorte que le message ne possède pas l'en-tête XQH attendu. Les messages doivent être insérés dans une file d'attente de transmission via une file d'attente éloignée, sauf si l'application génère l'en-tête MQXQH.

La valeur initiale de cette zone est MQRC\_NONE.

**DestQName (MQCHAR48) pour MQDLH**

DestQName est le nom de la file d'attente de messages qui était la destination d'origine du message.

La longueur de cette zone est indiquée par MQ\_Q\_NAME\_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

### **DestQMgrNom (MQCHAR48) pour MQDLH**

DestQMgrNom du gestionnaire de files d'attente qui était la destination d'origine du message.

La longueur de cette zone est indiquée par MQ\_Q\_MGR\_NAME\_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

### **Codage (MQLONG) pour MQDLH**

Le codage est le codage numérique des données qui suivent la structure MQDLH (généralement les données du message d'origine) ; il ne s'applique pas aux données numériques de la structure MQDLH elle-même.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données.

La valeur initiale de cette zone est 0.

### **CodedCharSetId (MQLONG) pour MQDLH**

CodedCharSetId est l'identificateur de jeu de caractères des données qui transitent par la structure MQDLH (généralement les données du message d'origine) ; il ne s'applique pas aux données de type caractère dans la structure MQDLH elle-même.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données. La valeur spéciale suivante peut être utilisée:

#### **MQCCSI\_HÉRITER**

Les données de type caractères des données qui suivent cette structure sont dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, la valeur MQCCSI\_INHERIT n'est pas renvoyée par l'appel MQGET.

Vous ne pouvez pas utiliser MQCCSI\_INHERIT si la valeur de la zone *PutApplType* dans MQMD est MQAT\_BROKER.

Cette valeur est prise en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour les clients IBM MQ connectés à ces systèmes.

La valeur initiale de cette zone est MQCCSI\_UNDEFINED.

### **Format (MQCHAR8) pour MQDLH**

Le format est le nom de format des données qui suivent la structure MQDLH (généralement les données du message d'origine).

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de codage de la zone *Format* dans MQMD.

La longueur de cette zone est indiquée par MQ\_FORMAT\_LENGTH. La valeur initiale de cette zone est MQFMT\_NONE.

### ***PutApplType (MQLONG) pour MQDLH***

PutApplType d'application qui place le message dans la file d'attente des messages non livrés.

Cette zone a la même signification que la zone *PutApplType* dans le descripteur de message MQMD (voir «MQMD-Descripteur de message», à la page 438 pour plus de détails).

Si le gestionnaire de files d'attente redirige le message vers la file d'attente de rebut, *PutApplType* a la valeur MQAT\_QMGR.

La valeur initiale de cette zone est 0.

### ***PutApplNom (MQCHAR28) pour MQDLH***

PutApplNom de l'application qui a inséré le message dans la file d'attente des messages non livrés.

Le format du nom dépend de la zone *PutApplType*. Le format peut varier d'une édition à l'autre. Voir la description de la zone *PutApplName* dans «MQMD-Descripteur de message», à la page 438.

Si le gestionnaire de files d'attente redirige le message vers la file d'attente de rebut, *PutApplName* contient les 28 premiers caractères du nom du gestionnaire de files d'attente, complétés par des blancs si nécessaire.

La longueur de cette zone est indiquée par MQ\_PUT\_APPL\_NAME\_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 28 caractères blancs dans les autres langages de programmation.

### ***PutDate (MQCHAR8) pour MQDLH***

PutDate est la date à laquelle le message a été inséré dans la file d'attente des messages non livrés.

Le format utilisé pour la date à laquelle cette zone est générée par le gestionnaire de files d'attente est le suivant:

- AAAAMMJJ

où les caractères représentent:

#### **AAAA**

année (quatre chiffres)

#### **MM**

mois de l'année (01 à 12)

#### **JJ**

jour du mois (01 à 31)

Le temps moyen de Greenwich (GMT) est utilisé pour les zones *PutDate* et *PutTime*, à condition que l'horloge système soit définie avec précision sur le temps moyen de Greenwich.

La longueur de cette zone est indiquée par MQ\_PUT\_DATE\_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et huit caractères blancs dans les autres langages de programmation.

### ***PutTime (MQCHAR8) pour MQDLH***

PutTime est l'heure à laquelle le message a été inséré dans la file d'attente des messages non livrés.

Le format utilisé pour l'heure à laquelle cette zone est générée par le gestionnaire de files d'attente est le suivant:

- HHMMSSSTH

où les caractères représentent:

#### **HH**

heures (00 à 23)

**MM**

minutes (00 à 59)

**SS**

secondes (00 à 59 ; voir remarque)

**T**

dixièmes de seconde (0 à 9)

**H**

centièmes de seconde (0 à 9)

**Remarque :** Si l'horloge système est synchronisée selon une norme de temps très précise, il est possible, dans de rares cas, que 60 ou 61 soient renvoyés pour les secondes dans *PutTime*. Cela se produit lorsque des secondes bissextiles sont insérées dans la norme de temps globale.

Le temps moyen de Greenwich (GMT) est utilisé pour les zones *PutDate* et *PutTime*, à condition que l'horloge système soit définie avec précision sur le temps moyen de Greenwich.

La longueur de cette zone est indiquée par MQ\_PUT\_TIME\_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et huit caractères blancs dans les autres langages de programmation.

## MQDMHO-Options de suppression de descripteur de message

La structure **MQDMHO** permet aux applications de spécifier des options qui contrôlent la façon dont les descripteurs de message sont supprimés. La structure est un paramètre d'entrée dans l'appel **MQDLTMH**.

### Jeu de caractères et codage

Les données dans **MQDMHO** doivent être dans le jeu de caractères de l'application et dans le codage de l'application (**MQENC\_NATIVE**).

### Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Tableau 486. Zones dans MQDMHO		
Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	MQDMHO_STRUC_ID	'DMHO'
<u>Version</u> (numéro de version de la structure)	MQDMHO_VERSION_1	1
<u>Options</u> (options)	MQDMHO_AUCUN	0
<p><b>Remarques :</b></p> <p>1. Dans le langage de programmation C, la variable macro MQDMHO_DEFAULT contient les valeurs répertoriées dans la table. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</p> <pre>MQDMHO MyDMHO = {MQDMHO_DEFAULT};</pre>		

### Déclarations de langue

Déclaration C pour MQDMHO

```
typedef struct tagMQDMHO;
struct tagMQDMHO {
```

```

MQCHAR4  StrucId;          /* Structure identifier */
MQLONG   Version;         /* Structure version number */
MQLONG   Options;         /* Options that control the action of MQDLTMH */
};

```

### Déclaration COBOL pour MQDMHO

```

**  MQDMHO structure
10  MQDMHO.
**  Structure identifier
15  MQDMHO-STRUCID      PIC X(4).
**  Structure version number
15  MQDMHO-VERSION     PIC S9(9) BINARY.
**  Options that control the action of MQDLTMH
15  MQDMHO-OPTIONS     PIC S9(9) BINARY.

```

### Déclaration PL/I pour MQDMHO

```

dcl
1  MQDMHO based,
3  StrucId      char(4),          /* Structure identifier */
3  Version      fixed bin(31),   /* Structure version number */
3  Options      fixed bin(31),   /* Options that control the action of MQDLTMH */

```

### Déclaration High Level Assembler pour MQDMHO

```

MQDMHO          DSECT
MQDMHO_STRUCID  DS   CL4   Structure identifier
MQDMHO_VERSION DS   F     Structure version number
MQDMHO_OPTIONS DS   F     Options that control the action of
*               MQDLTMH
MQDMHO_LENGTH  EQU   *-MQDMHO
MQDMHO_AREA    DS   CL(MQDMHO_LENGTH)

```

### **StrucId (MQCHAR4) pour MQDMHO**

Il s'agit de l'identificateur de structure de la structure d'options de suppression de descripteur de message. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQDMHO\_STRUC\_ID.

La valeur doit être:

#### **MQDMHO\_STRUC\_ID**

Identificateur de la structure d'options de suppression de descripteur de message.

Pour le langage de programmation C, la constante **MQDMHO\_STRUC\_ID\_ARRAY** est également définie. Il a la même valeur que **MQDMHO\_STRUC\_ID**, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQDMHO**

Il s'agit du numéro de version de la structure ; la valeur doit être:

#### **MQDMHO\_VERSION\_1**

Version-1 : supprime la structure d'options de descripteur de message.

La constante suivante indique le numéro de version de la version en cours:

#### **MQDMHO\_CURRENT\_VERSION**

Version actuelle de la structure d'options de suppression de descripteur de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **MQDMHO\_VERSION\_1**.

### **Options (MQLONG) pour MQDMHO**

La valeur doit être:

## **MQDMHO\_AUCUN**

Aucune option spécifiée.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **MQDMHO\_NONE**.

## **MQDMPO-Options de suppression de propriété de message**

La structure MQDMPO permet aux applications de spécifier des options qui contrôlent la façon dont les propriétés des messages sont supprimées. La structure est un paramètre d'entrée dans l'appel MQDLTMP.

### **Jeu de caractères et codage**

Les données de MQDMPO doivent être dans le jeu de caractères de l'application et dans le codage de l'application (MQENC\_NATIVE).

### **Zones**

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

<i>Tableau 487. Zones dans MQDMPO</i>		
<b>Nom et description de la zone</b>	<b>Nom de la constante</b>	<b>Valeur initiale (le cas échéant) de la constante</b>
<u>StrucId</u> (identificateur de structure)	ID_STRUC_MQDMPO_	' DMP0 '
<u>Version</u> (numéro de version de la structure)	MQDMPO_VERSION_1	1
<u>Options</u> (options contrôlant l'action de MQDMPO)	Options qui contrôlent l'action de MQDLTMP	MQDMPO_AUCUN
<b>Remarques :</b> 1. Dans le langage de programmation C, la variable macroMQDMPO_DEFAULT contient les valeurs répertoriées dans la table. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure: <pre>MQDMPO MyDMPO = {MQDMPO_DEFAULT};</pre>		

## **Déclarations de langue**

Déclaration C pour MQDMPO

```
typedef struct tagMQDMPO MQDMPO;
struct tagMQDMPO {
    MQCHAR4  StrucId;          /* Structure identifier */
    MQLONG   Version;         /* Structure version number */
    MQLONG   Options;         /* Options that control the action of
                               MQDLTMP */
};
```

Déclaration COBOL pour MQDMPO

```
**  MQDMPO structure
   10 MQDMPO.
**  Structure identifier
   15 MQDMPO-STRUCID          PIC X(4).
**  Structure version number
```

```

15 MQDMPO-VERSION          PIC S9(9) BINARY.
**  Options that control the action of MQDLTMP
15 MQDMPO-OPTIONS         PIC S9(9) BINARY.

```

## Déclaration PL/I pour MQDMPO

```

Dcl
  1 MQDMPO based,
  3 StrucId      char(4),      /* Structure identifier */
  3 Version      fixed bin(31), /* Structure version number */
  3 Options      fixed bin(31), /* Options that control the action
                                of MQDLTMP */

```

## Déclaration High Level Assembler pour MQDMPO

```

MQDMPO          DSECT
MQDMPO_STRUCID  DS   CL4  Structure identifier
MQDMPO_VERSION  DS   F    Structure version number
MQDMPO_OPTIONS  DS   F    Options that control the
*                  action of MQDLTMP
MQDMPO_LENGTH   EQU  *-MQDMPO
MQDMPO_AREA     DS   CL(MQDMPO_LENGTH)

```

### **StrucId (MQCHAR4) pour MQDMPO**

Il s'agit de l'identificateur de structure de la structure des options de suppression de propriété de message. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQDMPO\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUC\_MQDMPO\_**

Identificateur de la structure des options de suppression des propriétés de message.

Pour le langage de programmation C, la constante MQDMPO\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQDMPO\_STRUC\_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

### **Version (MQLONG) pour MQDMPO**

Supprimer la structure des options de propriété de message-Zone Version

Il s'agit du numéro de version de la structure. La valeur doit être:

#### **MQDMPO\_VERSION\_1**

Numéro de version de la structure des options de suppression de propriété de message.

La constante suivante indique le numéro de version de la version en cours:

#### **MQDMPO\_VERSION**

Version actuelle de la structure des options de suppression des propriétés de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQDMPO\_VERSION\_1.

### **Options (MQLONG) pour MQDMPO**

Supprimer la structure des options de propriété de message-Zone Options

**Options d'emplacement:** Les options suivantes sont liées à l'emplacement relatif de la propriété par rapport au curseur de propriété.

#### **MQDMPO\_DEL\_FIRST**

Supprime la première propriété qui correspond au nom spécifié.

#### **MQDMPO\_DEL\_PROP\_UNDER\_CURSOR**

Supprime la propriété pointée par le curseur de propriété ; il s'agit de la propriété qui a été interrogée pour la dernière fois à l'aide de l'option MQIMPO\_INQ\_FIRST ou MQIMPO\_INQ\_NEXT.



Le curseur de propriété est réinitialisé lorsque le descripteur de message est réutilisé. Il est également réinitialisé lorsque le descripteur de message est spécifié dans la zone *MsgHandle* de la structure MQGMO sur un appel MQGET ou MQPMO sur un appel MQPUT.

Si cette option est utilisée alors que le curseur de propriété n'a pas encore été établi, l'appel échoue avec le code achèvement MQCC\_FAILED et le code anomalie MQRC\_PROPERTY\_NOT\_AVAILABLE. Si la propriété pointée par le curseur de propriété a déjà été supprimée, l'appel échoue également avec le code achèvement MQCC\_FAILED et la raison MQRC\_PROPERTY\_NOT\_AVAILABLE.

Si aucune des options n'est requise, l'option suivante peut être utilisée:

#### **MQDMPO\_AUCUN**

Aucune option spécifiée.

Cette zone est toujours une zone d'entrée. La valeur initiale de cette zone est MQDMPO\_DEL\_FIRST.

## **MQEPH-en-tête PCF imbriqué**

La structure MQEPH décrit les données supplémentaires présentes dans un message lorsque ce message est un message PCF (Programmable Command Format). La zone *PCFHeader* définit les paramètres PCF qui suivent cette structure et permet de suivre les données de message PCF avec d'autres en-têtes.

### **Format**

MQFMT\_EMBEDDED\_PCF

### **Jeu de caractères et codage**

Les données de MQEPH doivent se trouver dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par MQENC\_NATIVE.

Définissez le jeu de caractères et le codage du MQEPH dans les zones *CodedCharSetId* et *Encoding* du MQMD (si la structure MQEPH est au début des données de message) ou dans la structure d'en-tête qui précède la structure MQEPH (tous les autres cas).

### **Utilisation**

Vous ne pouvez pas utiliser les structures MQEPH pour envoyer des commandes au serveur de commandes ou à tout autre serveur acceptant les PCF du gestionnaire de files d'attente.

De même, le serveur de commandes ou tout autre serveur d'acceptation PCF du gestionnaire de files d'attente ne génère pas de réponses ou d'événements contenant des structures MQEPH.

### **Zones**

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

<i>Tableau 488. Zones dans MQEPH pour MQEPH</i>		
<b>Nom et description de la zone</b>	<b>Nom de la constante</b>	<b>Valeur initiale (le cas échéant) de la constante</b>
<u>StrucId</u> (identificateur de structure)	ID_STRUC_MQEPH_ID	'EPH'
<u>Version</u> (numéro de version de la structure)	MQEPH_VERSION_1	1
<u>StrucLength</u> (longueur de la structure MQEPH plus les structures MQCFH et les structures de paramètres qui la suivent)	MQEPH_STRUC_LENGT H_FIXED	68

Tableau 488. Zones dans MQEPH pour MQEPH (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>Codage</u> (codage numérique des données qui suit la dernière structure de paramètres PCF)	Aucun	0
<u>CodedCharSetId</u> (identificateur de jeu de caractères des données qui suit la structure du dernier paramètre PCF)	MQCCSI_UNDEFINI	0
<u>Format</u> (nom de format des données qui suivent la dernière structure de paramètres PCF)	MQFMT_AUCUN	Espaces vides
<u>Indicateurs</u> (indicateurs)	MQEPH_NONE	0
<u>PCFHeader</u> (en-tête PCF)	Noms et valeurs définis dans <a href="#">Tableau 489</a> , à la <a href="#">page 381</a>	0
<p><b>Remarques :</b></p> <ol style="list-style-type: none"> <li>1. Le symbole ~ représente un caractère blanc unique.</li> <li>2. Dans le langage de programmation C, la variable macroMQEPH_DEFAULT contient les valeurs répertoriées dans la table. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</li> </ol> <pre style="background-color: #f0f0f0; padding: 10px;">MQEPH MyEPH = {MQEPH_DEFAULT};</pre>		

## Déclarations de langue

### Déclaration C pour MQEPH

```
typedef struct tagMQEPH MQEPH;
struct tagMQDH {
    MQCHAR4  StrucId;          /* Structure identifier */
    MQLONG   Version;        /* Structure version number */
    MQLONG   Struclength;     /* Total length of MQEPH including the MQCFH
                             and parameter structures that follow it */
    MQLONG   Encoding;       /* Numeric encoding of data that follows last
                             PCF parameter structure */
    MQLONG   CodedCharSetId; /* Character set identifier of data that
                             follows last PCF parameter structure */
    MQCHAR8  Format;          /* Format name of data that follows last PCF
                             parameter structure */
    MQLONG   Flags;           /* Flags */
    MQCFH    PCFHeader;      /* Programmable command format header */
};
```

### Déclaration COBOL pour MQEPH

```
** MQEPH structure
 10 MQEPH.
**   Structure identifier
 15 MQEPH-STRUCID      PIC X(4).
**   Structure version number
 15 MQEPH-VERSION     PIC S9(9) BINARY.
**   Total length of MQEPH structure including the MQCFH
**   and parameter structures that follow it
 15 MQEPH-STRUCLNGTH  PIC S9(9) BINARY.
**   Numeric encoding of data that follows last
**   PCF structure
 15 MQEPH-ENCODING    PIC S9(9) BINARY.
**   Character set identifier of data that
```

```

** follows last PCF parameter structure
15 MQEPH-CODEDCHARSETID PIC S9(9) BINARY.
** Format name of data that follows last PCF
** parameter structure
15 MQEPH-FORMAT PIC X(8).
** Flags
15 MQEPH-FLAGS PIC S9(9) BINARY.
** Programmable command format header
15 MQEPH-PCFHEADER.
** Structure type
20 MQEPH-PCFHEADER-TYPE PIC S9(9) BINARY.
** Structure length
20 MQEPH-PCFHEADER-STRULENGTH PIC S9(9) BINARY.
** Structure version number
20 MQEPH-PCFHEADER-VERSION PIC S9(9) BINARY.
** Command identifier
20 MQEPH-PCFHEADER-COMMAND PIC S9(9) BINARY.
** Message sequence number
20 MQEPH-PCFHEADER-MSGSEQNUMBER PIC S9(9) BINARY.
** Control options
20 MQEPH-PCFHEADER-CONTROL PIC S9(9) BINARY.
** Completion code
20 MQEPH-PCFHEADER-COMPCODE PIC S9(9) BINARY.
** Reason code qualifying completion code
20 MQEPH-PCFHEADER-REASON PIC S9(9) BINARY.
** Count of parameter structures
20 MQEPH-PCFHEADER-PARAMETERCOUNT PIC S9(9) BINARY.

```

### Déclaration PL/I pour MQEPH

```

dcl
  1 MQEPH based,
  3 StrucId char(4), /* Structure identifier */
  3 Version fixed bin(31), /* Structure version number */
  3 StrucLength fixed bin(31), /* Total Length of MQEPH including the
                               MQCFH and parameter structures that
                               follow it
  3 Encoding fixed bin(31), /* Numeric encoding of data that follows
                               last PCF parameter structure
  3 CodedCharSetId fixed bin(31), /* Character set identifier of data that
                               follows last PCF parameter structure
  3 Format char(8), /* Format name of data that follows last
                    PCF parameter structure */
  3 Flags fixed bin(31), /* Flags */
  3 PCFHeader, /* Programmable command format header
  5 Type fixed bin(31), /* Structure type */
  5 StrucLength fixed bin(31), /* Structure length */
  5 Version fixed bin(31), /* Structure version number */
  5 Command fixed bin(31), /* Command identifier */
  5 MsgseqNumber fixed bin(31), /* Message sequence number */
  5 Control fixed bin(31), /* Control options */
  5 CompCode fixed bin(31), /* Completion code */
  5 Reason fixed bin(31), /* Reason code qualifying completion code */
  5 ParameterCount fixed bin(31); /* Count of parameter structures */

```

### Déclaration High Level Assembler pour MQEPH

MQEPH	DSECT	
MQEPH_STRUCID	DS CL4	Structure identifier
MQEPH_VERSION	DS F	Structure version number
MQEPH_STRULENGTH	DS F	Total length of MQEPH including the
*		MQCFH and parameter structures that
		follow it
MQEPH_ENCODING	DS F	Numeric encoding of data that follows
*		last PCF parameter structure
MQEPH_CODEDCHARSETID	DS F	Character set identifier of data that
*		follows last PCF parameter structure
MQEPH_FORMAT	DS CL8	Format name of data that follows last
*		PCF parameter structure
MQEPH_FLAGS	DS F	Flags
MQEPH_PCFHEADER	DS 0F	Force fullword alignment
MQEPH_PCFHEADER_TYPE	DS F	Structure type
MQEPH_PCFHEADER_STRULENGTH	DS F	Structure length
MQEPH_PCFHEADER_VERSION	DS F	Structure version number
MQEPH_PCFHEADER_COMMAND	DS F	Command identifier
MQEPH_PCFHEADER_MSGSEQNUMBER	DS F	Structure length
MQEPH_PCFHEADER_CONTROL	DS F	Control options

MQEPH_PCFHEADER_COMPCODE	DS	F	Completion code
MQEPH_PCFHEADER_REASON	DS	F	Reason code qualifying completion code
MQEPH_PCFHEADER_PARAMETER COUNT	DS	F	Count of parameter structures
MQEPH_PCFHEADER_LENGTH	EQU	*	MQEPH_PCFHEADER
	ORG		MQEPH_PCFHEADER
MQEPH_PCFHEADER_AREA	DS		CL(MQEPH_PCFHEADER_LENGTH)
*			
MQEPH_LENGTH	EQU	*	MQEPH
	ORG		MQEPH
MQEPH_AREA	DS		CL(MQEPH_LENGTH)

## Déclaration Visual Basic pour MQEPH

```

Type MQEPH
  StrucId      As String*4 'Structure identifier'
  Version     As Long      'Structure version number'
  StrucLength  As Long      'Total length of MQEPH structure including the MQCFH'
                'and parameter structures that follow it'
  Encoding     As Long      'Numeric encoding of data that follows last'
                'PCF parameter structure'
  CodedCharSetId As Long    'Character set identifier of data that'
                'follows last PCF parameter structure'
  Format       As String*8 'Format name of data that follows last PCF'
                'parameter structure'
  Flags        As Long      'Flags'
  PCFHeader    As MQCFH    'Programmable command format header'
End Type

Global MQEPH_DEFAULT As MQEPH

```

### **StrucId (MQCHAR4) pour MQEPH**

Il s'agit de l'identificateur de structure de la structure d'en-tête PCF intégrée. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQEPH\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUC\_MQEPH\_ID**

Identificateur de la structure d'en-tête PCF intégrée.

Pour le langage de programmation C, la constante MQEPH\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQEPH\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQEPH**

La valeur doit être:

#### **MQEPH\_VERSION\_1**

Numéro de version de la structure d'en-tête PCF intégrée.

La constante suivante indique le numéro de version de la version en cours:

#### **MQCFH\_VERSION\_3**

Version actuelle de la structure d'en-tête PCF intégrée.

La valeur initiale de cette zone est MQEPH\_VERSION\_1.

### **StrucLength (MQLONG) pour MQEPH**

Il s'agit de la quantité de données précédant la structure d'en-tête suivante. Il comprend :

- Longueur de l'en-tête MQEPH
- Longueur de tous les paramètres PCF suivant l'en-tête
- Tout remplissage vide suivant ces paramètres

StrucLength doit être un multiple de 4.

La partie de longueur fixe de la structure est définie par MQEPH\_STRUC\_LENGTH\_FIXED.

La valeur initiale de cette zone est 68.

### **Codage (MQLONG) pour MQEPH**

Il s'agit du codage numérique des données qui suit la structure MQEPH et les paramètres PCF associés ; il ne s'applique pas aux données de type caractères de la structure MQEPH elle-même.

La valeur initiale de cette zone est 0.

### **CodedCharSetId (MQLONG) pour MQEPH**

Il s'agit de l'identificateur de jeu de caractères des données qui suivent la structure MQEPH et les paramètres PCF associés ; il ne s'applique pas aux données de type caractère de la structure MQEPH elle-même.

La valeur initiale de cette zone est MQCCSI\_UNDEFINED.

### **Format (MQCHAR8) pour MQEPH**

Il s'agit du nom de format des données qui suivent la structure MQEPH et les paramètres PCF associés.

La valeur initiale de cette zone est MQFMT\_NONE.

### **Indicateurs (MQLONG) pour MQEPH**

Les valeurs suivantes sont disponibles :

#### **MQEPH\_NONE**

Aucun indicateur n'a été spécifié. MQEPH\_NONE est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette constante soit utilisée avec d'autres, mais comme sa valeur est nulle, une telle utilisation ne peut pas être détectée.

#### **MQEPH\_CCSID\_IMBRIQUÉ**

Le jeu de caractères des paramètres contenant des données de type caractères est spécifié individuellement dans la zone CodedCharSetId de chaque structure. Le jeu de caractères des zones StrucId et Format est défini par la zone CodedCharSetId dans la structure d'en-tête qui précède la structure MQEPH, ou par la zone CodedCharSetId dans le MQMD si MQEPH se trouve au début du message.

La valeur initiale de cette zone est MQEPH\_NONE.

### **PCFHeader (MQCFH) pour MQEPH**

Il s'agit de l'en-tête PCF (Programmable Command Format) qui définit les paramètres PCF qui suivent la structure MQEPH. Cela vous permet de suivre les données de message PCF avec d'autres en-têtes.

L'en-tête PCF est initialement défini avec les valeurs suivantes:

<i>Tableau 489. Zones dans MQCFH</i>		
<b>Nom de zone</b>	<b>Nom de la constante</b>	<b>Valeur de la constante</b>
<i>Type</i>	MQCFT_AUCUN	0
<i>StrucLength</i>	LONGUEUR_STRUC_MQCFH	36
<i>Version</i>	MQCFH_VERSION_3	3
<i>StrucLength</i>	Aucun	0
<i>Command</i>	MQCMD_NONE	0
<i>MsgSeqNumber</i>	Aucun	1
<i>Control</i>	MQCFC_LAST	1
<i>CompCode</i>	MQCC_OK	0
<i>Reason</i>	MQRC_NONE	0

Tableau 489. Zones dans MQCFH (suite)		
Nom de zone	Nom de la constante	Valeur de la constante
<i>ParameterCount</i>	Aucun	0

L'application doit remplacer Type de MQCFT\_NONE par un type de structure valide pour l'utilisation de l'en-tête PCF imbriqué.

## Options MQGMO-Get-message

La structure MQGMO permet à l'application de contrôler la façon dont les messages sont supprimés des files d'attente. La structure est un paramètre d'entrée-sortie dans l'appel MQGET.

## Version

La version actuelle de MQGMO est MQGMO\_VERSION\_4. Certaines zones ne sont disponibles que dans certaines versions de MQGMO. Si vous devez porter des applications entre plusieurs environnements, vous devez vous assurer que la version de MQGMO est cohérente dans tous les environnements. Les zones qui existent uniquement dans des versions particulières de la structure sont identifiées comme telles dans «Options MQGMO-Get-message», à la page 382 et dans les descriptions de zone.

Les fichiers d'en-tête, COPY et INCLUDE fournis pour les langages de programmation pris en charge contiennent la version la plus récente de MQGMO prise en charge par l'environnement, mais avec la valeur initiale de la zone *Version* définie sur MQGMO\_VERSION\_1. Pour utiliser des zones qui ne sont pas présentes dans la structure version-1, définissez la zone *Version* sur le numéro de version de la version requise.

## Jeu de caractères et codage

Les données dans MQGMO doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et le codage du gestionnaire de files d'attente local indiqué par MQENC\_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Tableau 490. Zones de MQGMO pour MQGMO		
Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID_STRUC_MQGMO_	'GMO~'
<u>Version</u> (numéro de version de la structure)	MQGMO_VERSION_1	1
<u>Zone MQGMO-Options</u> (options qui contrôlent l'action de MQGET)	MQGMO_NO_WAIT	0
<u>WaitInterval</u> (intervalle d'attente)	Aucun	0
<u>Signal1</u> (signal)	Aucun	Pointeur null sur z/OS ; 0 dans le cas contraire
<u>Signal2</u> (identificateur de signal)	Aucun	0
<u>ResolvedQName</u> (nom résolu de la file d'attente de destination)	Aucun	Chaîne nulle ou blancs

Tableau 490. Zones de MQGMO pour MQGMO (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieur à MQGMO_VERSION_2.		
MatchOptions (options contrôlant les critères de sélection utilisés pour MQGET)	MQMO_MATCH_MSG_ID + MQMO_MATCH_CORREL_ID	3
GroupStatus (indicateur signalant si le message extrait se trouve dans un groupe)	MQGS_NOT_IN_GROUP E	'␣'
SegmentStatus (indicateur signalant si le message extrait est un segment d'un message logique)	MQSS_NON_SEGMENT	'␣'
Segmentation (indicateur signalant si une segmentation supplémentaire est autorisée pour le message extrait)	MQSEG_INHIBÉ	'␣'
Reserved1 (réservé)	Aucun	'␣'
<b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieur à MQGMO_VERSION_3.		
MsgToken (jeton de message)	MQMTOK_NONE	Valeurs NULL
ReturnedLength (longueur en octets des données de message renvoyées)	MQRL_UNDEFINED	-1
<b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieur à MQGMO_VERSION_4.		
Reserved2 (réservé)	Aucun	'␣'
MsgHandle (descripteur d'un message à remplir avec les propriétés du message extrait de la file d'attente)	MQHM_NONE	0
<p><b>Remarques :</b></p> <ol style="list-style-type: none"> <li>1. Le symbole ␣ représente un caractère blanc unique.</li> <li>2. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.</li> <li>3. Dans le langage de programmation C, la variable macro MQGMO_DEFAULT contient les valeurs répertoriées dans la table. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</li> </ol> <pre style="background-color: #f0f0f0; padding: 10px;">MQGMO MyGMO = {MQGMO_DEFAULT};</pre>		

## Déclarations de langue

Déclaration C pour MQGMO

```
typedef struct tagMQGMO MQGMO;
struct tagMQGMO {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    Options;          /* Options that control the action of */
                                /* MQGET */
```

```

MQLONG    WaitInterval;    /* Wait interval */
MQLONG    Signal1;         /* Signal */
MQLONG    Signal2;         /* Signal identifier */
MQCHAR48   ResolvedQName;  /* Resolved name of destination queue */
/* Ver:1 */
MQLONG    MatchOptions;    /* Options controlling selection */
/* criteria used for MQGET */
MQCHAR     GroupStatus;    /* Flag indicating whether message */
/* retrieved is in a group */
MQCHAR     SegmentStatus;  /* Flag indicating whether message */
/* retrieved is a segment of a logical */
/* message */
MQCHAR     Segmentation;   /* Flag indicating whether further */
/* segmentation is allowed for the */
/* message retrieved */
MQCHAR     Reserved1;      /* Reserved */
/* Ver:2 */
MQBYTE16   MsgToken;       /* Message token */
MQLONG     ReturnedLength; /* Length of message data returned */
/* (bytes) */
/* Ver:3 */
MQLONG     Reserved2;      /* Reserved */
MQHMSG     MsgHandle;      /* Message handle */
/* Ver:4 */
};

```

**Remarque :** Sous z/OS, la zone *Signal1* est déclarée en tant que PMQLONG.

Déclaration COBOL pour MQGMO

```

** MQGMO structure
10 MQGMO.
** Structure identifier
15 MQGMO-STRUCID PIC X(4).
** Structure version number
15 MQGMO-VERSION PIC S9(9) BINARY.
** Options that control the action of MQGET
15 MQGMO-OPTIONS PIC S9(9) BINARY.
** Wait interval
15 MQGMO-WAITINTERVAL PIC S9(9) BINARY.
** Signal
15 MQGMO-SIGNAL1 PIC S9(9) BINARY.
** Signal identifier
15 MQGMO-SIGNAL2 PIC S9(9) BINARY.
** Resolved name of destination queue
15 MQGMO-RESOLVEDQNAME PIC X(48).
** Options controlling selection criteria used for MQGET
15 MQGMO-MATCHOPTIONS PIC S9(9) BINARY.
** Flag indicating whether message retrieved is in a group
15 MQGMO-GROUPSTATUS PIC X.
** Flag indicating whether message retrieved is a segment of a
** logical message
15 MQGMO-SEGMENTSTATUS PIC X.
** Flag indicating whether further segmentation is allowed for the
** message retrieved
15 MQGMO-SEGMENTATION PIC X.
** Reserved
15 MQGMO-RESERVED1 PIC X.
** Message token
15 MQGMO-MSGTOKEN PIC X(16).
** Length of message data returned (bytes)
15 MQGMO-RETURNEDLENGTH PIC S9(9) BINARY.
** Reserved
15 MQGMO-RESERVED2 PIC S9(9) BINARY.
** Message handle
15 MQGMO-MSGHANDLE PIC S9(18) BINARY.

```

**Remarque :** Sous z/OS, la zone *Signal1* est déclarée en tant que POINTER.

Déclaration PL/I pour MQGMO

```

dcl
  1 MQGMO based,
    3 StrucId      char(4),          /* Structure identifier */
    3 Version     fixed bin(31),    /* Structure version number */
    3 Options     fixed bin(31),    /* Options that control the action of
                                   MQGET */

```



```

3 WaitInterval    fixed bin(31), /* Wait interval */
3 Signal1        fixed bin(31), /* Signal */
3 Signal2        fixed bin(31), /* Signal identifier */
3 ResolvedQName  char(48),      /* Resolved name of destination
                               queue */
3 MatchOptions   fixed bin(31), /* Options controlling selection
                               criteria used for MQGET */
3 GroupStatus    char(1),      /* Flag indicating whether message
                               retrieved is in a group */
3 SegmentStatus  char(1),      /* Flag indicating whether message
                               retrieved is a segment of a logical
                               message */
3 Segmentation   char(1),      /* Flag indicating whether further
                               segmentation is allowed for the
                               message retrieved */
3 Reserved1     char(1),      /* Reserved */
3 MsgToken       char(16),     /* Message token */
3 ReturnedLength fixed bin(31); /* Length of message data returned
                               (bytes) */
3 Reserved2     fixed bin(31); /* Reserved */
3 MsgHandle     fixed bin(63); /* Message handle */

```

**Remarque :** Sous z/OS, la zone *Signal1* est déclarée en tant que pointer.

#### Déclaration High Level Assembler pour MQGMO

```

MQGMO          DSECT
MQGMO_STRUCID  DS    CL4  Structure identifier
MQGMO_VERSION  DS    F    Structure version number
MQGMO_OPTIONS  DS    F    Options that control the action of
*              MQGET
MQGMO_WAITINTERVAL DS    F    Wait interval
MQGMO_SIGNAL1  DS    F    Signal
MQGMO_SIGNAL2  DS    F    Signal identifier
MQGMO_RESOLVEDQNAME DS    CL48 Resolved name of destination queue
MQGMO_MATCHOPTIONS DS    F    Options controlling selection criteria
*              used for MQGET
MQGMO_GROUPSTATUS DS    CL1  Flag indicating whether message
*              retrieved is in a group
MQGMO_SEGMENTSTATUS DS    CL1  Flag indicating whether message
*              retrieved is a segment of a logical
*              message
MQGMO_SEGMENTATION DS    CL1  Flag indicating whether further
*              segmentation is allowed for the message
*              retrieved
MQGMO_RESERVED1 DS    CL1  Reserved
MQGMO_MSGTOKEN  DS    XL16  Message token
MQGMO_RETURNEDLENGTH DS    F    Length of message data returned (bytes)
MQGMO_RESERVED2 DS    F    Reserved
MQGMO_MSGHANDLE DS    D    Message handle
MQGMO_LENGTH    EQU    *-MQGMO
MQGMO_AREA     DS    CL(MQGMO_LENGTH)

```

#### Déclaration High Level Assembler pour MQGMO

```

Type MQGMO
  StrucId      As String*4  'Structure identifier'
  Version      As Long      'Structure version number'
  Options      As Long      'Options that control the action of MQGET'
  WaitInterval As Long      'Wait interval'
  Signal1      As Long      'Signal'
  Signal2      As Long      'Signal identifier'
  ResolvedQName As String*48 'Resolved name of destination queue'
  MatchOptions As Long      'Options controlling selection criteria'
  GroupStatus  As String*1  'Flag indicating whether message'
  SegmentStatus As String*1 'retrieved is in a group'
  Segmentation As String*1  'Flag indicating whether message'
  Reserved1    As String*1  'retrieved is a segment of a logical'
  MsgToken     As MQBYTE16  'message'

```

ReturnedLength As Long  
End Type

'Length of message data returned (bytes)'

## Options de canal PROPCTL pour MQGMO

Utilisez l'attribut de canal **PROPCTL** pour contrôler les propriétés de message qui sont incluses dans un message envoyé d'un gestionnaire de files d'attente IBM MQ 9.3 à un gestionnaire de files d'attente partenaire à partir d'une version antérieure d' IBM MQ.

Tableau 491. Paramètres d'attribut de propriété de message de canal

PROPCTL	Description
tout	<p>Utilisez cette option si les applications connectées au gestionnaire de files d'attente partenaire à partir d'une version antérieure peuvent traiter les propriétés placées dans un message par une application IBM MQ 9.3 .</p> <p>Toutes les propriétés sont envoyées au gestionnaire de files d'attente partenaire, en plus des paires nom-valeur placées dans le MQRFH2.</p> <p>Vous devez prendre en compte deux problèmes liés à la conception de l'application :</p> <ol style="list-style-type: none"><li>1. Une application connectée au gestionnaire de files d'attente partenaire doit pouvoir traiter les messages contenant des en-têtes MQRFH2 générés sur un gestionnaire de files d'attente IBM MQ 9.3 .</li><li>2. L'application connectée au gestionnaire de files d'attente partenaire doit traiter correctement les nouvelles propriétés de message qui sont associées à MQPD_SUPPORT_REQUIRED.</li></ol> <p>Avec l'option de canal ALL définie, les applications JMS peuvent interagir entre IBM MQ 9.3 et une version antérieure à l'aide du canal. Les nouvelles applications IBM MQ 9.3 qui utilisent les propriétés de message peuvent interagir avec les applications d'une version antérieure, en fonction du mode de gestion des en-têtes MQRFH2 par l'application.</p>

Tableau 491. Paramètres d'attribut de propriété de message de canal (suite)

PROPCTL	Description
COMPAT	<p>Utilisez cette option pour envoyer des propriétés de message aux applications connectées à un gestionnaire de files d'attente dans certains cas uniquement. Les propriétés de message sont envoyées uniquement si deux conditions sont satisfaites :</p> <ol style="list-style-type: none"> <li>1. Aucune propriété ne doit être signalée comme nécessitant un traitement de propriété de message.</li> <li>2. Au moins l'une des propriétés de message doit être dans un dossier "réservé" ; voir <u>Remarque</u>.</li> </ol> <p>Avec l'option de canal COMPAT définie, les applications JMS peuvent interagir entre IBM MQ 9.3 et une version antérieure à l'aide du canal.</p> <p>Le canal n'est pas accessible à chaque application qui utilise les propriétés de message, il est disponible uniquement pour les applications qui utilisent les dossiers réservés. Les règles concernant l'envoi du message ou de la propriété sont les suivantes :</p> <ol style="list-style-type: none"> <li>1. Si le message a des propriétés, mais aucune d'entre elles n'est associée à un dossier "réservé", aucune propriété de message n'est envoyée.</li> <li>2. Si une propriété de message a été créée dans un dossier de propriété "réservé", toutes les propriétés de message associées au message sont envoyées. Toutefois : <ol style="list-style-type: none"> <li>a. Si les propriétés de message sont signalées comme nécessitant une prise en charge, MQPD_SUPPORT_REQUIRED ou MQPD_SUPPORT_REQUIRED_IF_LOCAL, le message est renvoyé dans son intégralité. Il est renvoyé, supprimé ou envoyé à la file d'attente de rebut en fonction de la valeur de ses options de rapport.</li> <li>b. Si aucune propriété de message n'est signalée comme nécessitant une prise en charge, il se peut qu'une propriété individuelle ne soit pas envoyée. Si les zones du descripteur de propriété de message sont paramétrées sur des valeurs autres que celles par défaut, la propriété individuelle n'est pas envoyée. Le message toujours envoyé. Un exemple de valeur de zone de descripteur de propriété autre que celle par défaut est MQPD_USER_CONTEXT.</li> </ol> </li> </ol> <p><b>Remarque :</b> Les noms de dossier "réservé" ont pour préfixe mcd., jms., usr. ou mqext.. Ces dossiers sont créés pour les applications qui utilisent l'interface JMS. Dans IBM MQ 9.3, toutes les paires nom-valeur placées dans ces dossiers sont traitées comme des propriétés de message.</p> <p>Les propriétés de message sont envoyées dans un en-tête MQRFH2, en plus des paires nom-valeur placées dans un en-tête MQRFH2. Toutes les paires nom-valeur placées dans un en-tête MQRFH2 sont envoyées tant que le message n'est pas rejeté.</p>
Aucun	<p>Utilisez cette option pour empêcher l'envoi des propriétés de message aux applications connectées à un gestionnaire de files d'attente partenaire d'une version antérieure. Un MQRFH2 contenant des paires nom-valeur et des propriétés de message est toujours envoyé, mais uniquement avec les paires nom-valeur.</p> <p>Une fois l'option de canal NONE définie, un message JMS est envoyé en tant que JMSTextMessage ou JMSBytesMessage sans propriété de message JMS. S'il est possible qu'une application de version antérieure ignore toutes les propriétés définies dans une application IBM MQ 9.3, elle peut interagir avec elle.</p>

### Options de file d'attente PROPCTL pour MQGMO

Utilisez l'attribut de file d'attente **PROPCTL** pour contrôler le renvoi des propriétés de message dans une application qui appelle **MQGET** sans définir d'option de propriété de message **MQGMO**.

Tableau 492. Paramètres d'attribut de propriété de message de file d'attente

PROPCTL	Description
tout	<p>Utilisez l'option ALL pour que les différentes applications qui lisent un message à partir de la même file d'attente puissent traiter le message de différentes manières.</p> <ul style="list-style-type: none"> <li>• Une application, qui a été migrée sans modification à partir d'une version antérieure peut poursuivre une lecture directe de MQRFH2. Les propriétés sont directement accessibles dans l'en-tête MQRFH2.</li> </ul> <p>Vous devez modifier l'application pour qu'elle gère de nouvelles propriétés et de nouveaux attributs de propriété. Il se peut que l'application soit affectée par les modifications au niveau de l'agencement et du nombre des en-têtes MQRFH2. Certains attributs de dossier peuvent être supprimés, ou IBM MQ signale une erreur dans la présentation de l'en-tête MQRFH2 qui a été ignorée dans une version antérieure.</p> <ul style="list-style-type: none"> <li>• Une application nouvelle ou modifiée peut utiliser la propriété de message MQI pour interroger les propriétés de message et lire directement les paires nom-valeur dans l'en-tête MQRFH2 .</li> </ul> <p>Toutes les propriétés dans le message sont renvoyées à l'application.</p> <ul style="list-style-type: none"> <li>• Si l'application appelle MQCRTMH pour créer un descripteur de message, elle doit analyser les propriétés de message à l'aide de MQINQMP. Les paires nom-valeur qui ne sont pas des propriétés de message restent dans le MQRFH2, qui est supprimé des propriétés de message.</li> <li>• Si l'application ne crée pas de descripteur de message, toutes les propriétés de message et les paires nom-valeur restent dans MQRFH2.</li> </ul> <p>ALL n'a cet effet que si l'application de réception n'a pas défini d'option MQGMO_PROPERTIES ou si elle l'a défini sur MQGMO_PROPERTIES_AS_Q_DEF.</p>

Tableau 492. Paramètres d'attribut de propriété de message de file d'attente (suite)

PROPCTL	Description
COMPAT (valeur par défaut)	<p>COMPAT est l'option par défaut. Si <code>GM0_PROPERTIES_*</code> n'est pas défini, comme dans une application de version antérieure non modifiée, l'option COMPAT est utilisée par défaut. En utilisant l'option COMPAT par défaut, une application d'une version antérieure qui n'a pas créé un en-tête MQRFH2 de manière explicite, fonctionne sans modification sur la IBM MQ 9.3.</p> <p>Utilisez cette option si vous avez écrit une application MQI d'une version antérieure pour la lecture de messages JMS.</p> <ul style="list-style-type: none"> <li>• Les propriétés JMS , qui sont stockées dans un en-tête MQRFH2 , sont renvoyées à l'application dans un en-tête MQRFH2 dans des dossiers dont les noms commencent par <code>mcd.</code>, <code>jms.</code>, <code>usr.</code> ou <code>mqext.</code></li> <li>• Si le message comporte des dossiers JMS et si une application IBM MQ 9.3 ajoute de nouveaux dossiers de propriétés au message, ces propriétés sont également renvoyées dans le fichier MQRFH2. Par conséquent, vous devez modifier l'application pour qu'elle gère de nouvelles propriétés et de nouveaux attributs de propriété. Il se peut qu'une application non modifiée soit affectée par les modifications au niveau de l'agencement et du nombre des en-têtes MQRFH2. Il est possible que certains attributs de dossier soient supprimés, ou IBM MQ détecte des erreurs dans la présentation de l'en-tête MQRFH2 ignorée dans une version antérieure.</li> </ul> <p><b>Remarque :</b> Dans ce scénario, le comportement de l'application est le même, qu'elle soit connectée à un gestionnaire de files d'attente de version antérieure ou IBM MQ 9.3. Si l'attribut <b>PROPCTL</b> du canal est paramétré sur COMPAT ou ALL, toutes les propriétés de message sont envoyées dans le message au gestionnaire de files d'attente partenaire de version antérieure.</p> <ul style="list-style-type: none"> <li>• Si le message n'est pas un message JMS, mais qu'il contient d'autres propriétés, ces propriétés ne sont pas renvoyées à l'application dans un en-tête MQRFH2.<sup>1</sup></li> <li>• L'option permet également aux applications de version antérieure qui créent de manière explicite un en-tête MQRFH2 de fonctionner correctement dans de nombreux cas. Par exemple, un programme MQI qui crée un en-tête MQRFH2 contenant des propriétés de message JMS continue à s'exécuter correctement. Si un message est créé sans propriété de message JMS mais avec d'autres dossiers MQRFH2, ces dossiers sont renvoyés à l'application. Seuls les dossiers de propriété de message sont supprimés de MQRFH2. Les dossiers de propriété de message sont identifiés grâce à leur nouvel attribut de dossier <code>content= 'properties'</code> , ou sont des dossiers dont les noms sont répertoriés dans <u>Nom de dossier de propriété définie</u> ou <u>Nom de dossier de propriété non groupée</u>.</li> <li>• Si l'application appelle MQCRTMH pour créer un descripteur de message, elle doit analyser les propriétés de message à l'aide de MQINQMP. Les propriétés de message sont supprimées des en-têtes MQRFH2. Les paires nom-valeur qui ne sont pas des propriétés de message restent dans MQRFH2.</li> <li>• Si l'application appelle MQCRTMH pour créer un descripteur de message, elle peut analyser toutes les propriétés de message, que le message comporte des dossiers JMS ou pas.</li> <li>• Si l'application ne crée pas de descripteur de message, toutes les propriétés de message et les paires nom-valeur restent dans MQRFH2.</li> </ul> <p>Si un message contient de nouveaux dossiers de propriété utilisateur, vous pouvez déduire que le message a été créé par une application IBM MQ 9.3 nouvelle ou modifiée. Si l'application réceptrice doit traiter ces nouvelles propriétés directement dans MQRFH2, vous devez la modifier pour utiliser l'option ALL. Une fois l'option COMPAT par défaut définie, une application non modifiée continue de traiter le reste de MQRFH2, sans les propriétés de la IBM MQ 9.3.</p> <p>L'objectif de l'interface PROPCTL est de prendre en charge les anciennes applications qui lisent les dossiers MQRFH2, ainsi que les nouvelles applications et les applications modifiées qui utilisent l'interface de propriété de message. Faites en sorte que les nouvelles applications puissent utiliser l'interface de propriété de message pour toutes</p>

Tableau 492. Paramètres d'attribut de propriété de message de file d'attente (suite)

PROPCTL	Description
FORCE	<p>L'option FORCE place toutes les propriétés de message dans les en-têtes MQRFH2. Toutes les propriétés de message et les paires nom-valeur des en-têtes MQRFH2 restent dans le message. Les propriétés de message ne sont pas supprimées de MQRFH2 et sont accessibles via un descripteur de message. La sélection de l'option FORCE permet à une application nouvellement migrée de lire les propriétés de message à partir des en-têtes MQRFH2.</p> <p>Supposons que vous avez modifié une application pour traiter les propriétés de message IBM MQ 9.3 tout en conservant sa capacité d'utiliser directement les en-têtes MQRFH2. Vous pouvez choisir quand l'application peut utiliser les propriétés de message en paramétrant l'attribut de file d'attente PROPCTL sur FORCE. Définissez l'attribut de file d'attente <b>PROPCTL</b> sur une autre valeur quand vous allez utiliser les propriétés de message. Si le comportement de la nouvelle fonction dans l'application n'est pas celui prévu, réattribuez la valeur FORCE à l'option <b>PROPCTL</b>.</p> <p>FORCE n'a cet effet que si l'application de réception n'a pas défini d'option MQGMO_PROPERTIES ou si elle l'a défini sur MQGMO_PROPERTIES_AS_Q_DEF.</p>
Aucun	<p>Utilisez l'option NONE pour qu'une application existante puisse traiter un message, en ignorant toutes les propriétés de message, et qu'une application nouvelle ou modifiée puisse interroger les propriétés de message.</p> <ul style="list-style-type: none"> <li>• Si l'application appelle MQCRTMH pour créer un descripteur de message, elle doit analyser les propriétés de message à l'aide de MQINQMP. Les paires nom-valeur qui ne sont pas des propriétés de message restent dans le MQRFH2, qui est supprimé des propriétés de message.</li> <li>• Si l'application ne crée pas un descripteur de message, toutes les propriétés de message sont supprimées de MQRFH2. Les paires nom-valeur dans les en-têtes MQRFH2 restent dans le message.</li> </ul> <p>NONE n'a cet effet que si l'application de réception n'a pas défini d'option MQGMO_PROPERTIES ou si elle l'a défini sur MQGMO_PROPERTIES_AS_Q_DEF.</p>
V6COMPAT	<p>Utilisez cette option pour recevoir MQRFH2 dans le même format dans lequel il a été envoyé. Si l'application émettrice ou le gestionnaire de files d'attente crée des propriétés de message additionnelles, ces dernières sont renvoyées dans le descripteur de message.</p> <p>Cette option doit être définie sur les files d'attente émettrices et réceptrices, et sur toutes les files d'attente de transmission intermédiaires. Elle remplace les autres options PROPCTL définies dans les définitions de file d'attente dans le chemin de résolution de nom de file d'attente.</p> <p>Utilisez l'option V6COMPAT uniquement dans des circonstances exceptionnelles. Par exemple, si vous migrez des applications d'une version antérieure vers IBM MQ 9.3, l'option est utile car elle préserve le comportement de la version antérieure. Cette option aura sans doute un impact sur le débit de messages. Elle est également plus difficile à gérer ; vous devez vous assurer que l'option est définie sur les files d'attente émettrices, réceptrices et de transmission intermédiaires.</p> <p>V6COMPAT n'a cet effet que si l'application de réception n'a pas défini d'option MQGMO_PROPERTIES ou si elle a défini cette option sur MQGMO_PROPERTIES_AS_Q_DEF.</p>

<sup>1</sup> L'existence de dossiers de propriétés spécifiques créés par IBM MQ classes for JMS indique un message JMS . Les dossiers de propriétés sont mcd . , jms . , usr . ou mqext .

Pour plus d'informations sur les propriétés de message et les paires nom-valeur, voir «NameValueData (MQCHARn) pour MQRFH2», à la page 558.

## Options de propriété de message pour MQGMO

Utilisez les options de propriété de message **MQGMO** pour contrôler la façon dont les propriétés de message sont renvoyées à une application.

<i>Tableau 493. Paramètres d'option de propriété de message MQGMO</i>	
<b>MQGMO Option</b>	<b>Description</b>
MQGMO_PROPERTIES_AS_Q_DEF	<p>Les applications IBM MQ qui lisent à partir de la même file d'attente et qui ne définissent pas <code>GMO_PROPERTIES_*</code> reçoivent les propriétés de message différemment. Les applications IBM MQ qui ne créent pas de descripteur de message sont contrôlées par l'attribut <b>PROPCTL</b> de la file d'attente. Une application IBM MQ peut choisir de recevoir des propriétés de message dans MQRFH2 ou de créer un descripteur de message et d'interroger les propriétés de message. Si l'application crée un descripteur de message, les propriétés sont supprimées de MQRFH2.</p> <ul style="list-style-type: none"> <li>• Une application IBM MQ nouvelle ou modifiée qui ne définit pas l'option <code>GMO_PROPERTIES_*</code> ou la définit sur <code>MQGMO_PROPERTIES_AS_Q_DEF</code>, peut analyser les propriétés de message. Cette application doit définir <code>MQCRTMH</code> pour créer un descripteur de message et analyser les propriétés de message à l'aide de l'appel <code>MQI MQINQMP</code>.</li> <li>• Si une application nouvelle ou modifiée ne crée pas de descripteur de message, elle doit lire les propriétés de message qu'elle reçoit directement des en-têtes MQRFH2.</li> <li>• Si l'attribut de file d'attente <b>PROPCTL</b> est défini sur <code>FORCE</code>, aucune propriété n'est renvoyée dans le descripteur de message. Toutes les propriétés sont renvoyées dans les en-têtes MQRFH2.</li> <li>• Si l'attribut de file d'attente <b>PROPCTL</b> est défini sur <code>NONE</code> ou <code>COMPAT</code>, une application IBM MQ qui crée un descripteur de message reçoit toutes les propriétés de message.</li> </ul>
MQGMO_PROPERTIES_IN_HANDLE	<p>Force une application à utiliser des propriétés de message. Utilisez cette option pour détecter si une application modifiée ne peut pas créer un descripteur de message. Il se peut que l'application tente de lire les propriétés de message directement à partir d'un en-tête MQRFH2, plutôt que d'appeler <code>MQINQMP</code>.</p>
MQGMO_NO_PROPERTIES	<ul style="list-style-type: none"> <li>• Toutes les propriétés sont supprimées. Les propriétés générées par le gestionnaire de files d'attente, telles que les propriétés JMS, sont supprimées.</li> <li>• Les propriétés sont supprimées même si le descripteur de message est créé. Les paires nom-valeur des autres dossiers MQRFH2 sont disponibles dans les données du message.</li> </ul>

Tableau 493. Paramètres d'option de propriété de message MQGMO (suite)

MQGMO Option	Description
MQGMO_PROPERTIES_FORCE_MQRFH2	<p>Les propriétés sont renvoyées dans les en-têtes MQRFH2, même si un descripteur de message est créé.</p> <ul style="list-style-type: none"> <li>• MQINQMP ne renvoie aucune propriété de message, même si le descripteur de message est créé. MQRC_PROPERTY_NOT_AVAILABLE est renvoyé si une propriété est analysée.</li> </ul>
MQGMO_PROPERTIES_COMPATIBILITY	<p>Si le message provient d'un client JMS, les propriétés JMS sont renvoyées dans les en-têtes MQRFH2. Les applications IBM MQ nouvelles ou modifiées qui créent un descripteur de message se comportent différemment.</p> <ul style="list-style-type: none"> <li>• Toutes les propriétés dans les dossiers de propriété de message sont renvoyées si le message comprend un dossier mcd., jms., usr. ou mqext.</li> <li>• Si le message contient des dossiers de propriété mais pas un dossier mcd., jms., usr. ou mqext, aucune propriété de message n'est renvoyée dans MQRFH2.</li> <li>• Si une application IBM MQ nouvelle ou modifiée crée un descripteur de message, analysez les propriétés de message à l'aide de l'appel MQI MQINQMP. Toutes les propriétés de message sont supprimées de MQRFH2.</li> <li>• Si une application IBM MQ nouvelle ou modifiée crée un descripteur de message, toutes les propriétés dans le message peuvent être analysées. Même si le message ne contient pas de dossier mcd., jms., usr. ou mqext, toutes les propriétés de message peuvent être analysées.</li> </ul>

**Référence associée**

PROPCTL

2471 (09A7) (RC2471) : MQRC\_PROPERTY\_NOT\_AVAILABLE

**StrucId (MQCHAR4) pour MQGMO**

Il s'agit de l'identificateur de structure de la structure des options d'obtention de message. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQGMO\_STRUC\_ID.

La valeur doit être:

**ID\_STRUC\_MQGMO\_**

Identificateur de la structure des options d'obtention de message.

Pour le langage de programmation C, la constante MQGMO\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQGMO\_STRUC\_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

**Version (MQLONG) pour MQGMO**

La version est le numéro de version de la structure.

Il doit s'agir de l'une des valeurs suivantes :

**MQGMO\_VERSION\_1**

Structure des options get-message Version-1 .

Cette version est prise en charge dans tous les environnements.



## **MQGMO\_VERSION\_2**

Structure des options d'obtention de message Version-2 .

Cette version est prise en charge dans tous les environnements.

## **MQGMO\_VERSION\_3**

Structure des options get-message Version-3 .

Cette version est prise en charge dans tous les environnements.

## **MQGMO\_VERSION\_4**

Structure des options get-message Version-4 .

Cette version est prise en charge dans tous les environnements.

Les zones qui existent uniquement dans les versions plus récentes de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

## **MQGMO\_CURRENT\_VERSION**

Version actuelle de la structure des options get-message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQGMO\_VERSION\_1.

## **Options (MQLONG) pour MQGMO**

Les options **MQGMO** contrôlent l'action de MQGET. Vous pouvez spécifier zéro ou plusieurs options. Si vous avez besoin de plusieurs valeurs facultatives:

- Ajoutez les valeurs (n'ajoutez pas la même constante plus d'une fois), ou
- Combinez les valeurs à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations bit).

Les combinaisons d'options non valides sont notées ; toutes les autres combinaisons sont valides.

## **Options d'attente**

Les options suivantes concernent l'attente de l'arrivée de messages dans la file d'attente:

### **MQGMO\_WAIT**

L'application attend l'arrivée d'un message approprié. La durée maximale d'attente de l'application est spécifiée dans *WaitInterval*.

**Important :** Il n'y a pas d'attente, ni de délai, si un message approprié est disponible immédiatement.

Si les demandes MQGET sont inhibées ou que les demandes MQGET sont inhibées pendant l'attente, l'attente est annulée. L'appel se termine avec MQCC\_FAILED et le code anomalie MQRC\_GET\_INHIBITED, qu'il y ait ou non des messages appropriés dans la file d'attente.

Vous pouvez utiliser MQGMO\_WAIT avec les options MQGMO\_BROWSE\_FIRST ou MQGMO\_BROWSE\_NEXT .

Si plusieurs applications sont en attente dans la même file d'attente partagée, les règles suivantes sélectionnent l'application qui est activée lorsqu'un message approprié arrive:

<i>Tableau 494. Règles d'activation des appels MQGET dans une file d'attente partagée.</i>		
<b>Nombre d'appels MQGET en attente d'activation</b>		<b>Résultat</b>
<b>Avec une option BROWSE</b>	<b>Sans option BROWSE<sup>2</sup></b>	
Aucun	Au moins un	Un appel MQGET sans option BROWSE est activé.

<sup>2</sup> Un appel MQGET spécifiant l'option MQGMO\_LOCK est traité comme un appel non navigant.

Tableau 494. Règles d'activation des appels MQGET dans une file d'attente partagée. (suite)


Nombre d'appels MQGET en attente d'activation		Résultat
Avec une option BROWSE	Sans option BROWSE <sup>2</sup>	
Au moins un	Aucun	Tous les appels MQGET avec une option BROWSE sont activés.
Au moins un	Au moins un	Un appel MQGET sans option BROWSE est activé. Le nombre d'appels MQGET avec une option BROWSE qui sont activés est imprévisible.

Si plusieurs appels MQGET sans option BROWSE sont en attente dans la même file d'attente, un seul appel est activé. Le gestionnaire de files d'attente tente d'accorder la priorité aux appels en attente dans l'ordre suivant:

1. Demandes get-wait spécifiques qui ne peuvent être satisfaites que par certains messages, par exemple, ceux avec un `MsgId` ou un `CorrelId` spécifique (ou les deux).
2. Demandes get-wait générales qui peuvent être satisfaites par n'importe quel message.

**Remarque :**

- Dans la première catégorie, aucune priorité supplémentaire n'est accordée aux demandes get-wait plus spécifiques. Par exemple, les demandes qui spécifient à la fois `MsgId` et `CorrelId`.
- Dans l'une ou l'autre catégorie, il n'est pas possible de prédire quelle application est sélectionnée. En particulier, l'application en attente la plus longue n'est pas nécessairement celle sélectionnée.
- La longueur du chemin et les considérations relatives à la planification de la priorité du système d'exploitation peuvent signifier qu'une application en attente dont la priorité du système d'exploitation est inférieure à celle attendue extrait le message.
- Il peut également arriver qu'une application qui n'est pas en attente récupère le message de préférence à un autre.

 Sous z/OS, les points suivants s'appliquent:

- Si vous souhaitez que l'application procède à d'autres travaux en attendant l'arrivée du message, envisagez d'utiliser l'option de signal (`MQGMO_SET_SIGNAL`) à la place. Toutefois, l'option de signal est spécifique à l'environnement ; les applications que vous devez porter entre différents environnements ne doivent pas l'utiliser.
- S'il y a plusieurs appels MQGET en attente du même message, avec un mélange d'options d'attente et de signal, chaque appel en attente est considéré de la même manière. Une erreur s'est produite lors de la spécification de `MQGMO_SET_SIGNAL` avec `MQGMO_WAIT`. Il est également erroné de spécifier cette option avec un descripteur de file d'attente pour lequel un signal est en attente.
- Si vous spécifiez `MQGMO_WAIT` ou `MQGMO_SET_SIGNAL` pour une file d'attente dont le `IndexType` est `MQIT_MSG_TOKEN`, aucun critère de sélection n'est autorisé. Ce qui signifie que :
  - Si vous utilisez un version-1 MQGMO, définissez les zones `MsgId` et `CorrelId` dans le MQMD spécifié dans l'appel MQGET à `MQMI_NONE` et `MQCI_NONE`.
  - Si vous utilisez une version version-2 ou ultérieure MQGMO, définissez la zone `MatchOptions` sur `MQMO_NONE`.
- Pour un appel MQGET sur une file d'attente partagée et que l'appel est une demande de navigation ou une extraction destructive d'un message de groupe, et que ni `MsgId` ni `CorrelId` ne doivent être mis en correspondance, votre ECB de signal est envoyé MQEC\_MSG\_ARRIVÉ après 200 millisecondes.

<sup>2</sup> Un appel MQGET spécifiant l'option `MQGMO_LOCK` est traité comme un appel non navigant.

Cela se produit, même si un message approprié peut ne pas être arrivé dans la file d'attente tant que l'intervalle d'attente n'est pas arrivé à expiration, lorsque la file d'attente est envoyée avec MQEC\_WAIT\_INTERVAL\_EXPIRED. Lorsque MQEC\_MSG\_ARRIVÉ est envoyé, vous devez réémettre un deuxième appel MQGET pour extraire le message, s'il est disponible.

Cette technique est utilisée pour vous assurer que vous êtes informé en temps utile de l'arrivée d'un message, mais elle peut apparaître comme une surcharge de traitement inattendue par rapport à une séquence d'appels similaire sur une file d'attente non partagée.

MQGMO\_WAIT est ignoré s'il est spécifié avec MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR ou MQGMO\_MSG\_UNDER\_CURSOR ; aucune erreur n'est générée.

### **MQGMO\_NO\_WAIT**

L'application n'attend pas si aucun message approprié n'est disponible. MQGMO\_NO\_WAIT est l'opposé de MQGMO\_WAIT. MQGMO\_NO\_WAIT est défini pour faciliter la documentation du programme. Il s'agit de la valeur par défaut si aucune n'est spécifiée.

### **MQGMO\_SET\_SIGNAL**

Utilisez cette option avec les zones Signal1 et Signal2 . Il permet aux applications de poursuivre avec d'autres travaux en attendant l'arrivée d'un message. Il permet également (si des fonctions de système d'exploitation appropriées sont disponibles) aux applications d'attendre que les messages arrivent dans plusieurs files d'attente.

**Remarque :** L'option MQGMO\_SET\_SIGNAL est spécifique à l'environnement ; ne l'utilisez pas pour les applications que vous souhaitez porter.

Dans deux cas, l'appel se termine de la même manière que si cette option n'avait pas été spécifiée:

1. Si un message actuellement disponible répond aux critères spécifiés dans le descripteur de message.
2. Si une erreur de paramètre ou une autre erreur synchrone est détectée.

Si aucun message répondant aux critères spécifiés dans le descripteur de message n'est actuellement disponible, le contrôle revient à l'application sans attendre l'arrivée d'un message. Les paramètres **CompCode** et **Reason** sont définis sur MQCC\_WARNING et MQRC\_SIGNAL\_REQUEST\_ACCEPTED. Les autres zones de sortie du descripteur de message et les paramètres de sortie de l'appel MQGET ne sont pas définis. Lorsqu'un message approprié arrive plus tard, le signal est délivré en déposant la BCE.

L'appelant doit ensuite émettre à nouveau l'appel MQGET pour extraire le message. L'application peut attendre ce signal, en utilisant des fonctions fournies par le système d'exploitation.

Si le système d'exploitation fournit un mécanisme d'attente multiple, vous pouvez l'utiliser pour attendre qu'un message arrive dans l'une des files d'attente.

Si un WaitInterval différent de zéro est spécifié, le signal est délivré après l'expiration de l'intervalle d'attente. Le gestionnaire de files d'attente peut également annuler l'attente, auquel cas le signal est fourni.

Plusieurs appels MQGET peuvent définir un signal pour le même message. L'ordre dans lequel les applications sont activées est le même que celui décrit pour MQGMO\_WAIT.

Si plusieurs appels MQGET attendent le même message, chaque appel en attente est considéré comme identique. Les appels peuvent inclure un mélange d'options d'attente et de signal.

Dans certaines conditions, l'appel MQGET peut extraire un message et un signal résultant de l'arrivée du même message peut être distribué. Lorsqu'un signal est délivré, une application doit être préparée pour qu'aucun message ne soit disponible.

Un descripteur de file d'attente ne peut pas comporter plus d'une demande de signal en attente.

Cette option n'est pas valide avec l'une des options suivantes:

- MQGMO\_UNLOCK
- MQGMO\_WAIT

Pour un appel MQGET dans une file d'attente partagée et que l'appel est une demande de navigation ou une extraction destructive d'un message de groupe, et que ni `MsgId` ni `CorrelId` ne doivent être mis en correspondance, le signal ECB de l'utilisateur est envoyé MQEC\_MSG\_ARRIVED après 200 millisecondes.

Cela se produit, même si un message approprié peut ne pas être arrivé dans la file d'attente tant que l'intervalle d'attente n'est pas arrivé à expiration, lorsque la file d'attente est envoyée avec MQEC\_WAIT\_INTERVAL\_EXPIRED. Lorsque MQEC\_MSG\_ARRIVED est envoyé, vous devez émettre à nouveau un deuxième appel MQGET pour extraire le message, le cas échéant.

Cette technique est utilisée pour vous assurer que vous êtes informé en temps utile de l'arrivée d'un message, mais elle peut apparaître comme une surcharge de traitement inattendue par rapport à une séquence d'appels similaire sur une file d'attente non partagée.

Il ne s'agit pas d'une méthode efficace d'extraction de messages lorsque les messages sont rarement ajoutés. Pour éviter cette surcharge pour le cas de navigation, spécifiez `MsgId` (s'il n'est pas indexé ou indexé par `MsgId`) ou `CorrelId` (s'il est indexé par `CorrelId`) correspondant à l'appel MQGET .

**z/OS** Cette option est prise en charge sous z/OS uniquement.

### **MQGMO\_FAIL\_IF QUIESCING**

Forcez l'échec de l'appel MQGET si le gestionnaire de files d'attente est à l'état de mise au repos.

**z/OS** Sous z/OS, cette option force également l'appel MQGET à échouer si la connexion (pour une application CICS ou IMS ) est à l'état de mise au repos.

Si cette option est spécifiée avec MQGMO\_WAIT ou MQGMO\_SET\_SIGNAL et que l'attente ou le signal est en attente au moment où le gestionnaire de files d'attente passe à l'état de mise au repos:

- L'attente est annulée et l'appel renvoie le code achèvement MQCC\_FAILED avec le code anomalie MQRC\_Q\_MGR QUIESCING ou MQRC\_CONNECTION QUIESCING.
- Le signal est annulé avec un code achèvement de signal spécifique à l'environnement.

**z/OS** Sous z/OS, le signal se termine par le code achèvement d'événement MQEC\_Q\_MGR QUIESCING ou MQEC\_CONNECTION QUIESCING.

Si MQGMO\_FAIL\_IF QUIESCING n'est pas spécifié et que le gestionnaire de files d'attente ou la connexion passe à l'état de mise au repos, l'attente ou le signal n'est pas annulé.

## **Options de point de synchronisation**

Les options suivantes concernent la participation de l'appel MQGET dans une unité d'oeuvre:

### **MQGMO\_SYNCPOINT**

La demande consiste à fonctionner dans le cadre des protocoles d'unité de travail normaux. Le message est marqué comme étant indisponible pour d'autres applications, mais il est supprimé de la file d'attente uniquement lorsque l'unité de travail est validée. Le message est de nouveau disponible si l'unité d'oeuvre est annulée.

Vous pouvez laisser MQGMO\_SYNCPOINT et MQGMO\_NO\_SYNCPOINT non définis. Dans ce cas, l'inclusion de la demande get dans les protocoles d'unité de travail est déterminée par l'environnement exécutant le gestionnaire de files d'attente. Il n'est pas déterminé par l'environnement qui exécute l'application.

- **z/OS** Sous z/OS, la demande d'obtention se trouve dans une unité de travail.
- Dans tous les environnements à l'exception de z/OS, la demande d'obtention ne se trouve pas dans une unité de travail.

En raison de ces différences, une application que vous souhaitez porter ne doit pas autoriser cette option à utiliser par défaut ; spécifiez explicitement MQGMO\_SYNCPOINT ou MQGMO\_NO\_SYNCPOINT .

Cette option n'est pas valide avec l'une des options suivantes:

- MQGMO\_BROWSE\_FIRST
- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
- MQGMO\_BROWSE\_NEXT
- MQGMO\_LOCK
- MQGMO\_NO\_SYNCPOINT
- MQGMO\_SYNCPOINT\_IF\_PERSISTENT
- MQGMO\_UNLOCK

### **MQGMO\_SYNCPOINT\_IF\_PERSISTENT**

La demande doit fonctionner dans les protocoles d'unité de travail normaux, mais uniquement si le message extrait est persistant. Un message persistant a la valeur MQPER\_PERSISTENT dans la zone Persistance de MQMD.

- Si le message est persistant, le gestionnaire de files d'attente traite l'appel comme si l'application avait spécifié MQGMO\_SYNCPOINT.
- Si le message n'est pas persistant, le gestionnaire de files d'attente traite l'appel comme si l'application avait spécifié MQGMO\_NO\_SYNCPOINT.

Cette option n'est pas valide avec l'une des options suivantes:

- MQGMO\_BROWSE\_FIRST
- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
- MQGMO\_BROWSE\_NEXT
- MQGMO\_COMPLETE\_MSG
- MQGMO\_MARK\_SKIP\_BACKOUT
- MQGMO\_NO\_SYNCPOINT
- MQGMO\_SYNCPOINT
- MQGMO\_UNLOCK

Cette option est prise en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  z/OS


et pour IBM MQ MQI clients connecté à ces systèmes.

### **MQGMO\_NO\_SYNCPOINT**

La demande consiste à fonctionner en dehors des protocoles d'unité de travail normaux. Si vous obtenez un message sans option de navigation, il est immédiatement supprimé de la file d'attente. Le message ne peut pas être rendu disponible à nouveau en raison de l'annulation de l'unité d'oeuvre.

Cette option est utilisée si vous spécifiez MQGMO\_BROWSE\_FIRST ou MQGMO\_BROWSE\_NEXT.

Vous pouvez laisser MQGMO\_SYNCPOINT et MQGMO\_NO\_SYNCPOINT non définis. Dans ce cas, l'inclusion de la demande get dans les protocoles d'unité de travail est déterminée par l'environnement exécutant le gestionnaire de files d'attente. Il n'est pas déterminé par l'environnement qui exécute l'application.

-  Sous z/OS, la demande d'obtention se trouve dans une unité de travail.
- Dans tous les environnements à l'exception de z/OS, la demande d'obtention ne se trouve pas dans une unité de travail.

En raison de ces différences, une application que vous souhaitez porter ne doit pas autoriser cette option à utiliser la valeur par défaut ; spécifiez explicitement MQGMO\_SYNCPOINT ou MQGMO\_NO\_SYNCPOINT .

Cette option n'est pas valide avec l'une des options suivantes:

- MQGMO\_MARK\_SKIP\_BACKOUT
- MQGMO\_SYNCPOINT
- MQGMO\_SYNCPOINT\_IF\_PERSISTENT

#### MQGMO\_MARK\_SKIP\_BACKOUT

Annulation d'une unité de travail sans réinstanciation dans la file d'attente du message marqué avec cette option.

Cette option est prise en charge uniquement sous z/OS.

Si cette option est spécifiée, MQGMO\_SYNCPOINT doit également être spécifié. MQGMO\_MARK\_SKIP\_BACKOUT n'est pas valide avec l'une des options suivantes:

- MQGMO\_BROWSE\_FIRST
- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
- MQGMO\_BROWSE\_NEXT
- MQGMO\_LOCK
- MQGMO\_NO\_SYNCPOINT
- MQGMO\_SYNCPOINT\_IF\_PERSISTENT
- MQGMO\_UNLOCK

**Remarque :** Sous IMS et CICS, vous pouvez être amené à émettre un appel IBM MQ extran après l'annulation d'une unité de travail contenant un message marqué avec MQGMO\_MARK\_SKIP\_BACKOUT. Vous devez émettre un appel IBM MQ avant de valider la nouvelle unité de travail contenant le message marqué. L'appel peut être n'importe quel appel IBM MQ de votre choix.

1. Sous IMS, si vous n'avez pas appliqué l' IMS APAR PN60855 et que vous exécutez une application IMS MPP ou BMP.
2. Sous CICS, si vous exécutez une application.

Dans les deux cas, émettez un appel IBM MQ avant de valider la nouvelle unité d'oeuvre contenant le message annulé.

**Remarque :** Dans une unité de travail, il ne peut y avoir qu'une seule demande d'obtention marquée comme annulation ignorée, ainsi qu'aucune ou plusieurs demandes d'obtention non marquées.

Si une application annule une unité de travail, un message qui a été extrait à l'aide de MQGMO\_MARK\_SKIP\_BACKOUT n'est pas restauré à son état précédent. Les autres mises à jour de ressource sont annulées. Le message est traité comme s'il avait été extrait dans une nouvelle unité de travail démarrée par la demande d'annulation. Le message est extrait sans l'option MQGMO\_MARK\_SKIP\_BACKOUT .

MQGMO\_MARK\_SKIP\_BACKOUT est utile si, après la modification de certaines ressources, il apparaît que l'unité de travail ne peut pas aboutir. Si vous omettez cette option, l'annulation de l'unité d'oeuvre rétablit le message dans la file d'attente. La même séquence d'événements se produit à nouveau, lors de la prochaine extraction du message.

Toutefois, si vous spécifiez MQGMO\_MARK\_SKIP\_BACKOUT sur l'appel MQGET d'origine, l'annulation de l'unité de travail annule les mises à jour apportées aux autres ressources. Le message est traité comme s'il avait été extrait sous une nouvelle unité de travail. L'application peut effectuer le traitement d'erreurs approprié. Il peut envoyer un message de rapport à l'expéditeur du message d'origine ou placer le message d'origine dans la file d'attente des messages non livrés. Il peut ensuite valider la nouvelle unité de travail. La validation de la nouvelle unité d'oeuvre supprime définitivement le message de la file d'attente d'origine.

MQGMO\_MARK\_SKIP\_BACKOUT marque un message physique unique. Si le message appartient à un groupe de messages, les autres messages du groupe ne sont pas marqués. De même, si le message marqué est un segment d'un message logique, les autres segments du message logique ne sont pas marqués.

Tout message d'un groupe peut être marqué, mais si des messages sont extraits à l'aide de MQGMO\_LOGICAL\_ORDER, il est avantageux de marquer le premier message du groupe. Si l'unité de travail est annulée, le premier message (marqué) est déplacé vers la nouvelle unité de travail. Le deuxième message et les messages suivants du groupe sont réintégrés dans la file d'attente. Les messages laissés dans la file d'attente ne peuvent pas être extraits par une autre application à l'aide de MQGMO\_LOGICAL\_ORDER. Le premier message du groupe n'est plus dans la file d'attente. Toutefois, l'application qui a sauvegardé l'unité de travail peut extraire le deuxième message et les messages ultérieurs dans la nouvelle unité de travail à l'aide de l'option MQGMO\_LOGICAL\_ORDER. Le premier message a déjà été extrait.

Il peut arriver que vous deviez revenir sur la nouvelle unité de travail. Par exemple, parce que la file d'attente de rebut est saturée et que le message ne doit pas être supprimé. L'annulation de la nouvelle unité d'oeuvre réintroduit le message dans la file d'attente d'origine, ce qui empêche la perte du message. Toutefois, dans cette situation, le traitement ne peut pas continuer. Après l'annulation de la nouvelle unité de travail, l'application doit informer l'opérateur ou l'administrateur qu'une erreur irrémédiable s'est produite, puis terminer.

MQGMO\_MARK\_SKIP\_BACKOUT ne fonctionne que si l'unité de travail contenant la demande get est interrompue par l'application qui l'a sauvegardé. Si l'unité de travail contenant la demande get est annulée en raison de l'échec de la transaction ou du système, MQGMO\_MARK\_SKIP\_BACKOUT est ignoré. Tout message extrait à l'aide de cette option est réintégré dans la file d'attente de la même manière que les messages extraits sans cette option.

## Options de navigation

Les options suivantes concernent la navigation dans les messages de la file d'attente:

### MQGMO\_BROWSE\_FIRST

Lorsqu'une file d'attente est ouverte avec l'option MQOO\_BROWSE, un curseur de navigation est établi, positionné logiquement avant le premier message de la file d'attente. Vous pouvez ensuite utiliser des appels MQGET en spécifiant l'option MQGMO\_BROWSE\_FIRST, MQGMO\_BROWSE\_NEXT ou MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR pour extraire les messages de la file d'attente de façon non destructive. Le curseur de navigation marque la position, dans les messages de la file d'attente, à partir de laquelle le prochain appel MQGET avec MQGMO\_BROWSE\_NEXT recherche un message approprié.

MQGMO\_BROWSE\_FIRST n'est pas valide avec l'une des options suivantes:

- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
- MQGMO\_BROWSE\_NEXT
- MQGMO\_MARK\_SKIP\_BACKOUT
- MQGMO\_MSG\_UNDER\_CURSOR
- MQGMO\_SYNCPOINT
- MQGMO\_SYNCPOINT\_IF\_PERSISTENT
- MQGMO\_UNLOCK

Il s'agit également d'une erreur si la file d'attente n'a pas été ouverte pour consultation.

Un appel MQGET avec MQGMO\_BROWSE\_FIRST ignore la position précédente du curseur de navigation. Le premier message de la file d'attente qui remplit les conditions spécifiées dans le descripteur de message est extrait. Le message reste dans la file d'attente et le curseur de navigation est positionné sur ce message.

Après cet appel, le curseur de navigation est positionné sur le message qui a été renvoyé. Le message peut être supprimé de la file d'attente avant l'émission du prochain appel MQGET avec

MQGMO\_BROWSE\_NEXT . Dans ce cas, le curseur de navigation reste à l'emplacement de la file d'attente occupé par le message, même si cette position est maintenant vide.

Utilisez l'option MQGMO\_MSG\_UNDER\_CURSOR avec un appel MQGET non-browse pour supprimer le message de la file d'attente.

Le curseur de navigation n'est pas déplacé par un appel MQGET non navigant, même si le même descripteur *Hobj* est utilisé. Elle n'est pas non plus déplacée par un appel browse MQGET qui renvoie le code achèvement MQCC\_FAILED ou le code anomalie MQRC\_TRUNCATED\_MSG\_FAILED.

Spécifiez l'option MQGMO\_LOCK avec cette option pour verrouiller le message consulté.

Vous pouvez spécifier MQGMO\_BROWSE\_FIRST avec n'importe quelle combinaison valide des options MQGMO\_\* et MQMO\_\* qui contrôlent le traitement des messages dans les groupes et les segments de messages logiques.

Si vous spécifiez MQGMO\_LOGICAL\_ORDER, les messages sont consultés dans l'ordre logique. Si vous omettez cette option, les messages sont parcourus dans l'ordre physique. Si vous spécifiez MQGMO\_BROWSE\_FIRST, vous pouvez basculer entre l'ordre logique et l'ordre physique. Les appels MQGET suivants utilisant MQGMO\_BROWSE\_NEXT parcourent la file d'attente dans le même ordre que l'appel le plus récent qui a spécifié MQGMO\_BROWSE\_FIRST pour l'identificateur de file d'attente.

Le gestionnaire de files d'attente conserve deux ensembles d'informations de groupe et de segment pour les appels MQGET . Les informations de groupe et de segment pour les appels de navigation sont conservées séparément des informations pour les appels qui suppriment les messages de la file d'attente. Si vous spécifiez MQGMO\_BROWSE\_FIRST, le gestionnaire de files d'attente ignore les informations de groupe et de segment à parcourir. Il analyse la file d'attente comme s'il n'existait aucun groupe en cours et aucun message logique en cours. Si l'appel MQGET aboutit, code achèvement MQCC\_OK ou MQCC\_WARNING, les informations de groupe et de segment à parcourir sont définies sur celles du message renvoyé. Si l'appel échoue, les informations de groupe et de segment restent les mêmes qu'avant l'appel.

### **MQGMO\_BROWSE\_NEXT**

Permet d'avancer le curseur de navigation jusqu'au message suivant de la file d'attente qui répond aux critères de sélection spécifiés dans l'appel MQGET . Le message est renvoyé à l'application, mais reste dans la file d'attente.

MQGMO\_BROWSE\_NEXT n'est pas valide avec l'une des options suivantes:

- MQGMO\_BROWSE\_FIRST
- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
- MQGMO\_MARK\_SKIP\_BACKOUT
- MQGMO\_MSG\_UNDER\_CURSOR
- MQGMO\_SYNCPOINT
- MQGMO\_SYNCPOINT\_IF\_PERSISTENT
- MQGMO\_UNLOCK

Il s'agit également d'une erreur si la file d'attente n'a pas été ouverte pour consultation.

MQGMO\_BROWSE\_NEXT se comporte de la même manière que MQGMO\_BROWSE\_FIRST, s'il s'agit du premier appel à parcourir une file d'attente, une fois que la file d'attente a été ouverte pour consultation.

Le message sous le curseur peut être supprimé de la file d'attente avant l'émission du prochain appel MQGET avec MQGMO\_BROWSE\_NEXT . Le curseur de navigation reste logiquement à l'emplacement de la file d'attente occupé par le message, même si cette position est maintenant vide.

Les messages sont stockés dans la file d'attente de deux manières:

- FIFO dans la priorité (MQMDS\_PRIORITY) ou
- FIFO quelle que soit la priorité (MQMDS\_FIFO)



L'attribut de file d'attente **MsgDeliverySequence** indique la méthode qui s'applique (voir «Attributs des files d'attente», à la page 875 pour plus de détails).

Une file d'attente peut avoir un `MsgDeliverySequence` de `MQMDS_PRIORITY`. Un message arrive dans la file d'attente dont la priorité est supérieure à celle indiquée par le curseur de navigation. Dans ce cas, le message de priorité supérieure est introuvable lors du balayage en cours de la file d'attente à l'aide de `MQGMO_BROWSE_NEXT`. Il ne peut être trouvé qu'une fois que le curseur de navigation a été réinitialisé avec `MQGMO_BROWSE_FIRST` ou en rouvrant la file d'attente.

L'option `MQGMO_MSG_UNDER_CURSOR` peut être utilisée avec un appel `MQGET` non-browse, si nécessaire, pour supprimer le message de la file d'attente.

Le curseur de navigation n'est pas déplacé par des appels `MQGET` non navigant utilisant le même descripteur `Hobj`.

Spécifiez l'option `MQGMO_LOCK` avec cette option pour verrouiller le message consulté.

Vous pouvez spécifier `MQGMO_BROWSE_NEXT` avec n'importe quelle combinaison valide des options `MQGMO_*` et `MQMO_*` qui contrôlent le traitement des messages dans les groupes et les segments de messages logiques.

Si vous spécifiez `MQGMO_LOGICAL_ORDER`, les messages sont consultés dans l'ordre logique. Si vous omettez cette option, les messages sont parcourus dans l'ordre physique. Si vous spécifiez `MQGMO_BROWSE_FIRST`, vous pouvez basculer entre l'ordre logique et l'ordre physique. Les appels `MQGET` suivants utilisant `MQGMO_BROWSE_NEXT` parcourent la file d'attente dans le même ordre que l'appel le plus récent qui a spécifié `MQGMO_BROWSE_FIRST` pour l'identificateur de file d'attente. L'appel échoue avec le code anomalie `MQRC_INCONSISTENT_BROWSE` si cette condition n'est pas satisfaite.

**Remarque :** Faites attention lors de l'utilisation d'un appel `MQGET` pour parcourir les données au-delà de la fin d'un groupe de messages si `MQGMO_LOGICAL_ORDER` n'est pas spécifié. Par exemple, supposons que le dernier message du groupe précède le premier message du groupe dans la file d'attente. L'utilisation de `MQGMO_BROWSE_NEXT` pour parcourir les données au-delà de la fin du groupe, en spécifiant `MQMO_MATCH_MSG_SEQ_NUMBER` avec `MsgSeqNumber` défini sur 1, renvoie le premier message du groupe déjà parcouru. Ce résultat peut se produire immédiatement ou un certain nombre d'appels `MQGET` par la suite s'il existe des groupes intermédiaires. La même remarque s'applique à un message logique qui ne se trouve pas dans un groupe.

Les informations de groupe et de segment pour les appels de navigation sont conservées séparément des informations pour les appels qui suppriment les messages de la file d'attente.

### **MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR**

Extrayez le message désigné par le curseur de navigation de manière non destructive, quelles que soient les options `MQMO_*` spécifiées dans la zone `MatchOptions` de `MQGMO`.

`MQGMO_BROWSE_MSG_UNDER_CURSOR` n'est pas valide avec l'une des options suivantes:

- `MQGMO_BROWSE_FIRST`
- `MQGMO_BROWSE_NEXT`
- `MQGMO_MARK_SKIP_BACKOUT`
- `MQGMO_MSG_UNDER_CURSOR`
- `MQGMO_SYNCPOINT`
- `MQGMO_SYNCPOINT_IF_PERSISTENT`
- `MQGMO_UNLOCK`

Il s'agit également d'une erreur si la file d'attente n'a pas été ouverte pour consultation.

Le message désigné par le curseur de navigation est celui qui a été extrait pour la dernière fois à l'aide de l'option `MQGMO_BROWSE_FIRST` ou `MQGMO_BROWSE_NEXT`. L'appel échoue si aucun de ces appels n'a été émis pour cette file d'attente depuis son ouverture. L'appel échoue également si le message qui se trouvait sous le curseur de navigation a depuis été extrait de façon destructive.

La position du curseur de navigation n'est pas modifiée par cet appel.

L'option `MQGMO_MSG_UNDER_CURSOR` peut être utilisée avec un appel `MQGET` non navigant, pour supprimer le message de la file d'attente.

Le curseur de navigation n'est pas déplacé par un appel `MQGET` non navigant, même si le même descripteur `Hobj` est utilisé. Elle n'est pas non plus déplacée par un appel `browse MQGET` qui renvoie le code achèvement `MQCC_FAILED` ou le code anomalie `MQRC_TRUNCATED_MSG_FAILED`.

Si `MQGMO_BROWSE_MSG_UNDER_CURSOR` est spécifié avec `MQGMO_LOCK`:

- Si un message est déjà verrouillé, il doit être celui qui se trouve sous le curseur, de sorte qu'il est renvoyé sans déverrouillage ni nouveau verrouillage. Le message reste verrouillé.
- S'il n'y a pas de message verrouillé et qu'il y a un message sous le curseur de navigation, il est verrouillé et renvoyé à l'application. S'il n'y a pas de message sous le curseur de navigation, l'appel échoue.

Si `MQGMO_BROWSE_MSG_UNDER_CURSOR` est spécifié sans `MQGMO_LOCK`:

- Si un message est déjà verrouillé, il doit être celui qui se trouve sous le curseur. Le message est renvoyé à l'application, puis déverrouillé. Etant donné que le message est maintenant déverrouillé, il n'est pas garanti qu'il puisse être consulté à nouveau ou extrait de façon destructive par la même application. Il se peut qu'il ait été extrait de façon destructive par une autre application qui extrait des messages de la file d'attente.
- S'il n'y a pas de message verrouillé et qu'il y a un message sous le curseur de navigation, il est renvoyé à l'application. S'il n'y a pas de message sous le curseur de navigation, l'appel échoue.

Si `MQGMO_COMPLETE_MSG` est spécifié avec `MQGMO_BROWSE_MSG_UNDER_CURSOR`, le curseur de navigation doit identifier un message dont la zone `Offset` dans `MQMD` est zéro. Si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie `MQRC_INVALID_MSG_UNDER_CURSOR`.

Les informations de groupe et de segment pour les appels de navigation sont conservées séparément des informations pour les appels qui suppriment les messages de la file d'attente.

### **MQGMO\_MSG\_UNDER\_CURSOR**

Extrayez le message désigné par le curseur de navigation, quelles que soient les options `MQMO_*` spécifiées dans la zone `MatchOptions` de `MQGMO`. Le message est supprimé de la file d'attente.

Le message désigné par le curseur de navigation est celui qui a été extrait pour la dernière fois à l'aide de l'option `MQGMO_BROWSE_FIRST` ou `MQGMO_BROWSE_NEXT`.

Si `MQGMO_COMPLETE_MSG` est spécifié avec `MQGMO_MSG_UNDER_CURSOR`, le curseur de navigation doit identifier un message dont la zone `Offset` dans `MQMD` est zéro. Si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie `MQRC_INVALID_MSG_UNDER_CURSOR`.

Cette option n'est pas valide avec l'une des options suivantes:

- `MQGMO_BROWSE_FIRST`
- `MQGMO_BROWSE_MSG_UNDER_CURSOR`
- `MQGMO_BROWSE_NEXT`
- `MQGMO_UNLOCK`

Il s'agit également d'une erreur si la file d'attente n'a pas été ouverte à la fois pour l'exploration et pour l'entrée. Si le curseur de navigation ne pointe pas vers un message récupérable, une erreur est renvoyée par l'appel `MQGET`.

### **MQGMO\_MARK\_BROWSE\_HANDLE**

Le message renvoyé par un `MQGET` réussi, ou identifié par le `MsgToken` renvoyé, est marqué. La marque est spécifique au descripteur d'objet utilisé dans l'appel.

Le message n'est pas supprimé de la file d'attente.

`MQGMO_MARK_BROWSE_HANDLE` est valide uniquement si l'une des options suivantes est également spécifiée:

- `MQGMO_BROWSE_FIRST`

- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
- MQGMO\_BROWSE\_NEXT

MQGMO\_MARK\_BROWSE\_HANDLE n'est pas valide avec l'une des options suivantes:

- MQGMO\_ALL\_MSGS\_AVAILABLE
- MQGMO\_ALL\_SEGMENTS\_AVAILABLE
- MQGMO\_COMPLETE\_MSG
- MQGMO\_LOCK
- MQGMO\_LOGICAL\_ORDER
- MQGMO\_UNLOCK

Le message reste dans cet état jusqu'à ce que l'un des événements suivants se produise:

- La poignée d'objet concernée est fermée, normalement ou non.
- Le message n'est pas marqué pour ce descripteur par un appel à MQGET avec l'option MQGMO\_UNMARK\_BROWSE\_HANDLE.
- Le message est renvoyé à partir d'un appel à MQGET destructif, qui se termine par MQCC\_OK ou MQCC\_WARNING. L'état du message reste modifié même si le MQGET est ultérieurement annulé.
- Le message expire.

### **MQGMO\_MARK\_BROWSE\_CO\_OP**

Le message renvoyé par un MQGET réussi ou identifié par le *MsgToken* renvoyé est marqué pour tous les descripteurs de l'ensemble coopérant.

La marque de niveau coopératif s'ajoute à toute marque de niveau de descripteur qui aurait pu être définie.

Le message n'est pas supprimé de la file d'attente.

MQGMO\_MARK\_BROWSE\_CO\_OP est valide uniquement si le descripteur d'objet utilisé a été renvoyé par un appel à MQOPEN qui a spécifié MQOO\_CO\_OP. Vous devez également spécifier l'une des options MQGMO suivantes:

- MQGMO\_BROWSE\_FIRST
- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
- MQGMO\_BROWSE\_NEXT

Cette option n'est pas valide avec l'une des options suivantes:

- MQGMO\_ALL\_MSGS\_AVAILABLE
- MQGMO\_ALL\_SEGMENTS\_AVAILABLE
- MQGMO\_COMPLETE\_MSG
- MQGMO\_LOCK
- MQGMO\_LOGICAL\_ORDER
- MQGMO\_UNLOCK

Si le message est déjà marqué et que l'option MQGMO\_UNMARKED\_BROWSE\_MSG n'est pas spécifiée, l'appel échoue avec MQCC\_FAILED et le code anomalie MQRC\_MSG\_MARKED\_BROWSE\_CO\_OP.

Le message reste dans cet état jusqu'à ce que l'un des événements suivants se produise:

- Tous les descripteurs d'objet de l'ensemble coopérant sont fermés.
- Le message n'est pas marqué pour les navigateurs associés par un appel à MQGET avec l'option MQGMO\_UNMARK\_BROWSE\_CO\_OP.
- Le message est automatiquement démarqué par le gestionnaire de files d'attente.
- Le message est renvoyé à partir d'un appel à un MQGET non navigant. L'état du message reste modifié même si le MQGET est ultérieurement annulé.

- Le message expire.

#### **MQGMO\_UNMARKED\_BROWSE\_MSG**

Un appel à MQGET qui spécifie MQGMO\_UNMARKED\_BROWSE\_MSG renvoie un message qui est considéré comme non marqué pour son descripteur. Il ne renvoie pas de message si le message a été marqué pour son descripteur. Il ne renvoie pas non plus le message si la file d'attente a été ouverte par un appel à MQOPEN, avec l'option MQOO\_CO\_OP, et que le message a été marqué par un membre de l'ensemble de coopération.

Cette option n'est pas valide avec l'une des options suivantes:

- MQGMO\_ALL\_MSGS\_AVAILABLE
- MQGMO\_ALL\_SEGMENTS\_AVAILABLE
- MQGMO\_COMPLETE\_MSG
- MQGMO\_LOCK
- MQGMO\_LOGICAL\_ORDER
- MQGMO\_UNLOCK

#### **MQGMO\_UNMARK\_BROWSE\_CO\_OP**

Après un appel à MQGET qui spécifie cette option, le message n'est plus pris en compte par les descripteurs ouverts du jeu de descripteurs coopérants à marquer pour le jeu de descripteurs coopérants. Le message est toujours considéré comme marqué au niveau de la poignée s'il a été marqué au niveau de la poignée avant cet appel.

L'utilisation de MQGMO\_UNMARK\_BROWSE\_CO\_OP est valide uniquement avec un descripteur renvoyé par un appel réussi à MQOPEN avec l'option MQOO\_CO\_OP. Le MQGET aboutit même si le message n'est pas considéré comme étant marqué par l'ensemble de descripteurs coopérant.

MQGMO\_UNMARK\_BROWSE\_CO\_OP n'est pas valide sur un appel MQGET non-browse, ou avec l'une des options suivantes:

- MQGMO\_ALL\_MSGS\_AVAILABLE
- MQGMO\_ALL\_SEGMENTS\_AVAILABLE
- MQGMO\_COMPLETE\_MSG
- MQGMO\_LOCK
- MQGMO\_LOGICAL\_ORDER
- MQGMO\_MARK\_BROWSE\_CO\_OP
- MQGMO\_UNLOCK
- MQGMO\_UNMARKED\_BROWSE\_MSG

#### **MQGMO\_UNMARK\_BROWSE\_HANDLE**

Après un appel à MQGET qui spécifie cette option, le message localisé n'est plus considéré comme étant marqué par cet identificateur.

L'appel aboutit même si le message n'est pas marqué pour ce descripteur.

Cette option n'est pas valide sur un appel MQGET non-browse ou avec l'une des options suivantes:

- MQGMO\_ALL\_MSGS\_AVAILABLE
- MQGMO\_ALL\_SEGMENTS\_AVAILABLE
- MQGMO\_COMPLETE\_MSG
- MQGMO\_LOCK
- MQGMO\_LOGICAL\_ORDER
- MQGMO\_MARK\_BROWSE\_CO\_OP
- MQGMO\_UNLOCK
- MQGMO\_UNMARKED\_BROWSE\_MSG

## Options de verrouillage

Les options suivantes concernent le verrouillage des messages dans la file d'attente:

### MQGMO\_LOCK

Verrouillez le message qui est parcouru, de sorte que le message devienne invisible pour tout autre descripteur ouvert pour la file d'attente. L'option ne peut être spécifiée que si l'une des options suivantes est également spécifiée:

- MQGMO\_BROWSE\_FIRST
- MQGMO\_BROWSE\_NEXT
- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR

Un seul message peut être verrouillé pour chaque descripteur de file d'attente. Il peut s'agir d'un message logique ou d'un message physique:

- Si vous spécifiez MQGMO\_COMPLETE\_MSG, tous les segments de message qui constituent le message logique sont verrouillés sur le descripteur de file d'attente. Les messages doivent tous être présents dans la file d'attente et disponibles pour extraction.
- Si vous omettez MQGMO\_COMPLETE\_MSG, un seul message physique est verrouillé pour l'identificateur de file d'attente. Si ce message se trouve être un segment d'un message logique, le segment verrouillé empêche les autres applications utilisant MQGMO\_COMPLETE\_MSG d'extraire ou de parcourir le message logique.

Le message verrouillé est toujours celui qui se trouve sous le curseur de navigation. Le message peut être supprimé de la file d'attente par un appel MQGET ultérieur qui spécifie l'option MQGMO\_MSG\_UNDER\_CURSOR. D'autres appels MQGET utilisant le descripteur de file d'attente peuvent également supprimer le message (par exemple, un appel qui spécifie l'identificateur du message verrouillé).

Si l'appel renvoie le code achèvement MQCC\_FAILED ou MQCC\_WARNING avec le code anomalie MQRC\_TRUNCATED\_MSG\_FAILED, aucun message n'est verrouillé.

Si l'application ne supprime pas le message de la file d'attente, le verrou est libéré par l'une des actions suivantes:

- Emission d'un autre appel MQGET pour cet indicateur, en spécifiant MQGMO\_BROWSE\_FIRST ou MQGMO\_BROWSE\_NEXT. Le verrou est libéré si l'appel se termine avec MQCC\_OK ou MQCC\_WARNING. Le message reste verrouillé si l'appel se termine par MQCC\_FAILED. Cependant, les exceptions suivantes s'appliquent :
  - Le message n'est pas déverrouillé si MQCC\_WARNING est renvoyé avec MQRC\_TRUNCATED\_MSG\_FAILED.
  - Le message est déverrouillé si MQCC\_FAILED est renvoyé avec MQRC\_NO\_MSG\_AVAILABLE.

Si vous spécifiez également MQGMO\_LOCK, le message renvoyé est verrouillé. Si vous omettez MQGMO\_LOCK, aucun message n'est verrouillé après l'appel.

Si vous spécifiez MQGMO\_WAIT et qu'aucun message n'est immédiatement disponible, le message d'origine est déverrouillé avant le début de l'attente.

- Emission d'un autre appel MQGET pour ce descripteur, avec MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR, sans MQGMO\_LOCK. Le verrou est libéré si l'appel se termine avec MQCC\_OK ou MQCC\_WARNING. Le message reste verrouillé si l'appel se termine par MQCC\_FAILED. Toutefois, l'exception suivante s'applique:
  - Le message n'est pas déverrouillé si MQCC\_WARNING est renvoyé avec MQRC\_TRUNCATED\_MSG\_FAILED.
- Emission d'un autre appel MQGET pour ce descripteur avec MQGMO\_UNLOCK.
- Emission d'un appel MQCLOSE à l'aide de l'indicateur. MQCLOSE peut être implicite, en raison de l'arrêt de l'application.

Aucune option spéciale MQOPEN n'est requise pour spécifier MQGMO\_LOCK, autre que MQOO\_BROWSE, qui est nécessaire pour spécifier une option de navigation associée.

MQGMO\_LOCK n'est pas valide avec l'une des options suivantes:

- MQGMO\_MARK\_SKIP\_BACKOUT
- MQGMO\_SYNCPOINT
- MQGMO\_SYNCPOINT\_IF\_PERSISTENT
- MQGMO\_UNLOCK

### **MQGMO\_UNLOCK**

Le message à déverrouiller doit avoir été préalablement verrouillé par un appel MQGET avec l'option MQGMO\_LOCK. Si aucun message n'est verrouillé pour cet identificateur, l'appel se termine avec MQCC\_WARNING et MQRC\_NO\_MSG\_LOCKED.

Les paramètres **MsgDesc**, **BufferLength**, **Buffer** et **DataLength** ne sont pas vérifiés ou modifiés si vous spécifiez MQGMO\_UNLOCK. Aucun message n'est renvoyé dans *Buffer*.

Aucune option d'ouverture spéciale n'est requise pour spécifier MQGMO\_UNLOCK (bien que MQOO\_BROWSE soit nécessaire pour émettre la demande de verrouillage en premier lieu).

Cette option n'est valide qu'avec les options suivantes:

- MQGMO\_NO\_WAIT
- MQGMO\_NO\_SYNCPOINT

Ces deux options sont supposées être spécifiées ou non.

## **Options de données de message**

Les options suivantes concernent le traitement des données de message lorsque le message est lu à partir de la file d'attente:

### **MQGMO\_ACCEPT\_TRUNCATED\_MSG**

Si la mémoire tampon de messages est trop petite pour contenir le message complet, autorisez l'appel MQGET à remplir la mémoire tampon. MQGET remplit la mémoire tampon avec autant de messages que possible. Il émet un code achèvement d'avertissement et termine son traitement. Ce qui signifie que :

- Lors de la navigation dans les messages, le curseur de navigation est avancé jusqu'au message renvoyé.
- Lors de la suppression de messages, le message renvoyé est supprimé de la file d'attente.
- Le code anomalie MQRC\_TRUNCATED\_MSG\_ACCEPTED est renvoyé si aucune autre erreur ne se produit.

Sans cette option, la mémoire tampon est toujours remplie avec autant de messages qu'elle peut contenir. Un code achèvement d'avertissement est émis, mais le traitement n'est pas terminé. Ce qui signifie que :

- Lors de la navigation dans les messages, le curseur de navigation n'est pas avancé.
- Lors de la suppression de messages, le message n'est pas supprimé de la file d'attente.
- Le code anomalie MQRC\_TRUNCATED\_MSG\_FAILED est renvoyé si aucune autre erreur ne se produit.

### **MQGMO\_CONVERT**

Cette option convertit les données d'application du message pour qu'elles soient conformes aux valeurs CodedCharSetId et Encoding spécifiées dans le paramètre **MsgDesc** de l'appel MQGET. Les données sont converties avant d'être copiées dans le paramètre **Buffer**.

La zone Format spécifiée lors de l'insertion du message est prise en compte par le processus de conversion pour identifier la nature des données du message. Les données de message sont converties par le gestionnaire de files d'attente pour les formats intégrés et par un exit écrit par

l'utilisateur pour les autres formats. Pour plus de détails sur l'exit de conversion de données, voir «[Exit de conversion de données](#)», à la page 951.

- Si la conversion aboutit, les zones `CodedCharSetId` et `Encoding` spécifiées dans le paramètre **MsgDesc** sont inchangées lors du retour de l'appel `MQGET`.
- Si seule la conversion échoue, les données de message sont renvoyées non converties. Les zones `CodedCharSetId` et `Encoding` de `MsgDesc` sont définies sur les valeurs du message non converti. Le code achèvement est `MQCC_WARNING` dans ce cas.

Dans les deux cas, ces zones décrivent l'identificateur de jeu de caractères et le codage des données de message renvoyées dans le paramètre **Buffer**.

Voir la zone *Format* décrite dans «[MQMD-Descripteur de message](#)», à la page 438 pour la liste des noms de format pour lesquels le gestionnaire de files d'attente effectue la conversion.

## Options de groupe et de segment

Les options suivantes concernent le traitement des messages dans des groupes et des segments de messages logiques. Avant les descriptions d'option, voici quelques définitions des termes importants:

### Message physique

Un message physique est la plus petite unité d'informations pouvant être placée dans une file d'attente ou supprimée d'une file d'attente. Il correspond souvent aux informations spécifiées ou extraites sur un seul appel `MQPUT`, `MQPUT1` ou `MQGET`. Chaque message physique possède son propre descripteur de message, `MQMD`. En règle générale, les messages physiques se distinguent par des valeurs différentes pour l'identificateur de message, la zone `MsgId` dans `MQMD`. Le gestionnaire de files d'attente n'impose pas de valeurs différentes.

### Message logique

Un message logique est une unité unique d'informations d'application. En l'absence de contraintes système, un message logique est identique à un message physique. Si les messages logiques sont volumineux, les contraintes du système peuvent rendre souhaitable ou nécessaire la division d'un message logique en deux ou plusieurs messages physiques, appelés segments.

Un message logique segmenté se compose d'au moins deux messages physiques ayant le même identificateur de groupe non null, zone `GroupId` dans `MQMD`. Ils ont le même numéro de séquence de message, zone `MsgSeqNumber` dans `MQMD`. Les segments se distinguent par des valeurs différentes pour le décalage de segment, zone `Offset` dans `MQMD`. Le décalage de segment est le décalage des données du message physique à partir du début des données du message logique. Chaque segment étant un message physique, les segments d'un message logique ont généralement des identificateurs de message différents.

Un message logique qui n'a pas été segmenté, mais pour lequel la segmentation a été autorisée par l'application émettrice, possède également un identificateur de groupe non null. Dans ce cas, il n'y a qu'un seul message physique avec cet identificateur de groupe si le message logique n'appartient pas à un groupe de messages. Les messages logiques, pour lesquels la segmentation a été inhibée par l'application émettrice, ont un identificateur de groupe nul, `MQGI_NONE`, sauf si le message logique appartient à un groupe de messages.

### Groupe de messages

Un groupe de messages est un ensemble d'un ou de plusieurs messages logiques ayant le même identificateur de groupe non null. Les messages logiques du groupe se distinguent par des valeurs différentes pour le numéro de séquence de message. Le numéro de séquence est un entier compris entre 1 et *n*, où *n* est le nombre de messages logiques dans le groupe. Si un ou plusieurs des messages logiques sont segmentés, il y a plus de *n* messages physiques dans le groupe.

### MQGMO\_LOGICAL\_ORDER

`MQGMO_LOGICAL_ORDER` contrôle l'ordre dans lequel les messages sont renvoyés par les appels `MQGET` successifs pour le descripteur de file d'attente. L'option doit être spécifiée à chaque appel.

Si `MQGMO_LOGICAL_ORDER` est spécifié pour des appels `MQGET` successifs pour le même descripteur de file d'attente, les messages des groupes sont renvoyés dans l'ordre de leur numéro de séquence de message. Les segments de messages logiques sont renvoyés dans l'ordre indiqué par leurs

décalages de segment. Cet ordre peut être différent de l'ordre dans lequel ces messages et segments apparaissent dans la file d'attente.

**Remarque :** La spécification de MQGMO\_LOGICAL\_ORDER n'a aucune conséquence négative sur les messages qui n'appartiennent pas à des groupes et qui ne sont pas des segments. En effet, ces messages sont traités comme si chacun appartenait à un groupe de messages composé d'un seul message. Il est prudent de spécifier MQGMO\_LOGICAL\_ORDER lors de l'extraction de messages à partir de files d'attente qui contiennent un mélange de messages dans des groupes, des segments de message et des messages non segmentés dans des groupes.

Pour renvoyer les messages dans l'ordre requis, le gestionnaire de files d'attente conserve les informations de groupe et de segment entre les appels MQGET successifs. Les informations de groupe et de segment identifient le groupe de messages en cours et le message logique en cours pour l'identificateur de file d'attente. Il identifie également la position en cours dans le groupe et le message logique, et indique si les messages sont extraits dans une unité d'oeuvre. Etant donné que le gestionnaire de files d'attente conserve ces informations, l'application n'a pas besoin de définir les informations de groupe et de segment avant chaque appel MQGET . En particulier, cela signifie que l'application n'a pas besoin de définir les zones GroupId, MsgSeqNumber et Offset dans MQMD. Toutefois, l'application doit définir correctement l'option MQGMO\_SYNCPOINT ou MQGMO\_NO\_SYNCPOINT à chaque appel.

Lorsque la file d'attente est ouverte, il n'y a pas de groupe de messages en cours ni de message logique en cours. Un groupe de messages devient le groupe de messages en cours lorsqu'un message comportant l'indicateur MQMF\_MSG\_IN\_GROUP est renvoyé par l'appel MQGET . Si MQGMO\_LOGICAL\_ORDER est spécifié lors d'appels successifs, ce groupe reste le groupe en cours jusqu'à ce qu'un message contenant:

- MQMF\_LAST\_MSG\_IN\_GROUP sans MQMF\_SEGMENT (c'est-à-dire que le dernier message logique du groupe n'est pas segmenté), ou
- MQMF\_LAST\_MSG\_IN\_GROUP avec MQMF\_LAST\_SEGMENT (c'est-à-dire que le message renvoyé est le dernier segment du dernier message logique du groupe).

Lorsqu'un tel message est renvoyé, le groupe de messages est arrêté et, à la fin de l'appel à MQGET , il n'y a plus de groupe en cours. De la même manière, un message logique devient le message logique en cours lorsqu'un message comportant l'indicateur MQMF\_SEGMENT est renvoyé par l'appel MQGET . Le message logique est arrêté lorsque le message comportant l'indicateur MQMF\_LAST\_SEGMENT est renvoyé.

Si aucun critère de sélection n'est spécifié, les appels MQGET successifs renvoient, dans l'ordre correct, les messages du premier groupe de messages de la file d'attente. Ils renvoient ensuite les messages pour le deuxième groupe de messages, et ainsi de suite, jusqu'à ce qu'il n'y ait plus de messages disponibles. Il est possible de sélectionner les groupes de messages particuliers renvoyés en spécifiant une ou plusieurs des options suivantes dans la zone MatchOptions :

- MQMO\_MATCH\_MSG\_ID
- MQMO\_MATCH\_CORREL\_ID
- MQMO\_MATCH\_GROUP\_ID

Toutefois, ces options ne sont effectives que lorsqu'il n'y a pas de groupe de messages ou de message logique en cours. Voir la zone MatchOptions décrite dans [«Options MQGMO-Get-message»](#), à la page 382 pour plus de détails.

Tableau 495, à la page 409 affiche les valeurs des zones MsgId, CorrelId, GroupId, MsgSeqNumber et Offset que le gestionnaire de files d'attente recherche lorsqu'il tente de trouver un message à renvoyer lors de l'appel MQGET . Les règles s'appliquent à la fois à la suppression des messages de la file d'attente et à la consultation des messages de la file d'attente. Dans le tableau, l'un ou l'autre signifie Oui ou Non:

#### **LOG ORD**

Indique si l'option MQGMO\_LOGICAL\_ORDER est spécifiée sur l'appel.



**Cur grp**

Indique si un groupe de messages en cours existe avant l'appel.

**Cur log msg**

Indique si un message logique en cours existe avant l'appel.

**Autres colonnes**

Affiche les valeurs que le gestionnaire de files d'attente recherche. Précédent indique la valeur renvoyée pour la zone dans le message précédent pour le descripteur de file d'attente.

Tableau 495. Options MQGET relatives aux messages dans les groupes et les segments de messages logiques

Options que vous spécifiez	Statut du groupe et du message de journal avant l'appel		Valeurs que le gestionnaire de files d'attente recherche				
	Cur grp	Cur log msg	MsgId	CorrelId	GroupId	MsgSeqNumber	Offset
LOG ORD	Cur grp	Cur log msg	MsgId	CorrelId	GroupId	MsgSeqNumber	Offset
Oui	Non	Non	Contrôlé par MatchOptions	Contrôlé par MatchOptions	Contrôlé par MatchOptions	1	0
Oui	Non	Oui	Tout identificateur de message	Tout identificateur de corrélation	Identificateur de groupe précédent	1	Décalage précédent + longueur de segment précédent
Oui	Oui	Non	Tout identificateur de message	Tout identificateur de corrélation	Identificateur de groupe précédent	Numéro de séquence précédent + 1	0
Oui	Oui	Oui	Tout identificateur de message	Tout identificateur de corrélation	Identificateur de groupe précédent	Numéro de séquence précédent	Décalage précédent + longueur de segment précédent
Non	L'un ou l'autre	L'un ou l'autre	Contrôlé par MatchOptions	Contrôlé par MatchOptions	Contrôlé par MatchOptions	Contrôlé par MatchOptions	Contrôlé par MatchOptions

Si plusieurs groupes de messages sont présents dans la file d'attente et peuvent être renvoyés, ils sont renvoyés dans l'ordre déterminé par la position dans la file d'attente du premier segment du premier message logique de chaque groupe. En d'autres termes, les messages physiques dont les numéros de séquence sont 1 et les décalages 0 déterminent l'ordre dans lequel les groupes admissibles sont renvoyés.

L'option MQGMO\_LOGICAL\_ORDER affecte les unités de travail comme suit:

- Si le premier message logique ou segment d'un groupe est extrait dans une unité d'oeuvre, tous les autres messages logiques et segments du groupe doivent être extraits dans une unité d'oeuvre, si le même descripteur de file d'attente est utilisé. Toutefois, il n'est pas nécessaire de les extraire dans la même unité de travail. Cela permet à un groupe de messages composé de plusieurs messages physiques d'être scindé en plusieurs unités de travail consécutives pour l'identificateur de file d'attente.
- Si le premier message logique ou segment d'un groupe n'est pas extrait dans une unité de travail et que le même descripteur de file d'attente est utilisé, aucun des autres messages logiques et segments du groupe ne peut être extrait dans une unité de travail.

Si ces conditions ne sont pas satisfaites, l'appel MQGET échoue avec le code anomalie MQRC\_INCONSISTENT\_UOW.

Lorsque MQGMO\_LOGICAL\_ORDER est spécifié, le MQGMO fourni dans l'appel MQGET ne doit pas être inférieur à MQGMO\_VERSION\_2 et le MQMD ne doit pas être inférieur à MQMD\_VERSION\_2. Si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie MQRC\_WRONG\_GMO\_VERSION ou MQRC\_WRONG\_MD\_VERSION, selon le cas.

Si MQGMO\_LOGICAL\_ORDER n'est pas spécifié pour les appels MQGET successifs de l'identificateur de file d'attente, les messages sont renvoyés indépendamment du fait qu'ils appartiennent à des groupes de messages ou qu'ils soient des segments de messages logiques. Cela signifie que les messages ou les segments d'un groupe ou d'un message logique particulier peuvent être renvoyés dans l'ordre, ou mélangés avec des messages ou des segments d'autres groupes ou messages logiques, ou avec des messages qui ne sont pas dans des groupes et qui ne sont pas des segments. Dans ce cas, les messages particuliers renvoyés par les appels MQGET successifs sont contrôlés par les options MQMO\_\* spécifiées sur ces appels (voir la zone *MatchOptions* décrite dans «Options MQGMO-Get-message», à la page 382 pour plus de détails sur ces options).

Il s'agit de la technique qui peut être utilisée pour redémarrer un groupe de messages ou un message logique au milieu, après une défaillance du système. Lorsque le système redémarre, l'application peut définir les zones GroupId, MsgSeqNumber, Offset et MatchOptions sur les valeurs appropriées, puis émettre l'appel MQGET avec MQGMO\_SYNCPOINT ou MQGMO\_NO\_SYNCPOINT défini, mais sans spécifier MQGMO\_LOGICAL\_ORDER. Si cet appel aboutit, le gestionnaire de files d'attente conserve les informations de groupe et de segment, et les appels MQGET suivants utilisant cet identificateur de file d'attente peuvent spécifier MQGMO\_LOGICAL\_ORDER comme normal.

Les informations de groupe et de segment que le gestionnaire de files d'attente conserve pour l'appel MQGET sont distinctes des informations de groupe et de segment qu'il conserve pour l'appel MQPUT. En outre, le gestionnaire de files d'attente conserve des informations distinctes pour:

- MQGET appels qui suppriment des messages de la file d'attente.
- MQGET appels qui parcourent les messages de la file d'attente.

Pour un descripteur de file d'attente donné, l'application peut combiner des appels MQGET qui spécifient MQGMO\_LOGICAL\_ORDER avec des appels MQGET qui ne le font pas. Notez toutefois les points suivants:

- Si vous omettez MQGMO\_LOGICAL\_ORDER, chaque appel MQGET réussi entraîne le gestionnaire de files d'attente à définir les informations de groupe et de segment sauvegardées sur les valeurs correspondant au message renvoyé ; cette opération remplace les informations de groupe et de segment existantes conservées par le gestionnaire de files d'attente pour l'identificateur de file d'attente. Seules les informations appropriées à l'action de l'appel (parcourir ou supprimer) sont modifiées.
- Si vous omettez MQGMO\_LOGICAL\_ORDER, l'appel n'échoue pas s'il existe un groupe de messages en cours ou un message logique ; l'appel peut aboutir avec un code achèvement MQCC\_WARNING . Le Tableau 496, à la page 411 montre les différentes observations qui peuvent survenir. Dans ces cas, si le code achèvement n'est pas MQCC\_OK, le code anomalie est l'un des suivants (selon le cas):
  - MQRC\_INCOMPLETE\_GROUP
  - MQRC\_INCOMPLETE\_MSG
  - MQRC\_INCONSISTENT\_UOW

**Remarque :** Le gestionnaire de files d'attente ne vérifie pas les informations de groupe et de segment lors de la consultation d'une file d'attente ou lors de la fermeture d'une file d'attente qui a été ouverte pour consultation mais pas d'entrée ; dans ces cas, le code achèvement est toujours MQCC\_OK (en supposant qu'il n'y a pas d'autres erreurs).

Tableau 496. Résultat lorsque l'appel MQGET ou MQCLOSE n'est pas cohérent avec les informations de groupe et de segment

L'appel en cours est	L'appel précédent était MQGET avec MQGMO_LOGICAL_ORDER	L'appel précédent était MQGET sans MQGMO_LOGICAL_ORDER
MQGET avec MQGMO_LOGICAL_ORDER	MQCC_FAILED	MQCC_FAILED
MQGET sans MQGMO_LOGICAL_ORDER	MQCC_WARNING	MQCC_OK
MQCLOSE avec un groupe non terminé ou un message logique	MQCC_WARNING	MQCC_OK

Les applications qui souhaitent extraire des messages et des segments dans l'ordre logique sont recommandées pour spécifier MQGMO\_LOGICAL\_ORDER, car il s'agit de l'option la plus simple à utiliser. Cette option dispense l'application de la nécessité de gérer les informations de groupe et de segment, car le gestionnaire de files d'attente gère ces informations. Cependant, les applications spécialisées peuvent avoir besoin de plus de contrôle que celui fourni par l'option MQGMO\_LOGICAL\_ORDER, et cela peut être réalisé en ne spécifiant pas cette option. L'application doit ensuite s'assurer que les zones MsgId, CorrelId, GroupId, MsgSeqNumber et Offset dans MQMD, et les options MQMO\_\* dans MatchOptions dans MQGMO, sont correctement définies, avant chaque appel MQGET.

Par exemple, une application qui souhaite réacheminer les messages physiques qu'elle reçoit, indépendamment du fait que ces messages se trouvent dans des groupes ou des segments de messages logiques, ne doit pas spécifier MQGMO\_LOGICAL\_ORDER. Dans un réseau complexe avec plusieurs chemins entre les gestionnaires de files d'attente d'envoi et de réception, les messages physiques peuvent arriver dans le désordre. En spécifiant ni MQGMO\_LOGICAL\_ORDER, ni le MQPMO\_LOGICAL\_ORDER correspondant sur l'appel MQPUT, l'application de réacheminement peut extraire et réacheminer chaque message physique dès qu'il arrive, sans avoir à attendre le message suivant dans l'ordre logique d'arrivée.

Vous pouvez spécifier MQGMO\_LOGICAL\_ORDER avec l'une des autres options MQGMO\_\* et avec diverses options MQMO\_\* dans les circonstances appropriées (voir la section précédente).

- **z/OS** Sous z/OS, cette option est prise en charge pour les files d'attente privées et partagées, mais la file d'attente doit avoir le type d'index MQIT\_GROUP\_ID. Pour les files d'attente partagées, l'objet CFSTRUCT que la file d'attente mappe doit être au niveau CFLEVEL (3) ou supérieur.

- Cette option est prise en charge pour toutes les files d'attente locales pour les plateformes suivantes:

- **AIX** AIX
- **Linux** Linux
- **IBM i** IBM i
- **Windows** Windows

et pour IBM MQ MQI clients connecté à ces systèmes,.

### MQGMO\_COMPLETE\_MSG

Seul un message logique complet peut être renvoyé par l'appel MQGET. Si le message logique est segmenté, le gestionnaire de files d'attente réassemble les segments et renvoie le message logique complet à l'application. Le fait que le message logique ait été segmenté n'est pas évident pour l'application qui l'extrait.

**Remarque :** Il s'agit de la seule option qui permet au gestionnaire de files d'attente de réassembler des segments de message. S'ils ne sont pas spécifiés, les segments sont renvoyés individuellement

à l'application s'ils sont présents dans la file d'attente (et répondent aux autres critères de sélection spécifiés dans l'appel MQGET ). Les applications qui ne souhaitent pas recevoir de segments individuels doivent toujours spécifier MQGMO\_COMPLETE\_MSG.

Pour utiliser cette option, l'application doit fournir une mémoire tampon suffisamment grande pour contenir le message complet, ou spécifier l'option MQGMO\_ACCEPT\_TRUNCATED\_MSG .

Si la file d'attente contient des messages segmentés dont certains des segments sont manquants (peut-être parce qu'ils ont été retardés dans le réseau et qu'ils ne sont pas encore arrivés), la spécification de MQGMO\_COMPLETE\_MSG empêche l'extraction de segments appartenant à des messages logiques incomplets. Toutefois, ces segments de message contribuent toujours à la valeur de l'attribut de file d'attente **CurrentQDepth** ; cela signifie qu'il peut n'y avoir aucun message logique récupérable, même si *CurrentQDepth* est supérieur à zéro.

Pour les messages persistants, le gestionnaire de files d'attente peut réassembler les segments uniquement dans une unité de travail:

- Si l'appel MQGET est effectué dans une unité de travail définie par l'utilisateur, cette unité de travail est utilisée. Si l'appel échoue lors du processus de réassemblage, le gestionnaire de files d'attente rétablit dans la file d'attente tous les segments qui ont été supprimés lors du réassemblage. Toutefois, l'échec n'empêche pas la validation de l'unité de travail.
- Si l'appel est effectué en dehors d'une unité de travail définie par l'utilisateur et qu'il n'existe aucune unité de travail définie par l'utilisateur, le gestionnaire de files d'attente crée une unité de travail pour la durée de l'appel. Si l'appel aboutit, le gestionnaire de files d'attente valide automatiquement l'unité d'oeuvre (l'application n'a pas besoin de le faire). Si l'appel échoue, le gestionnaire de files d'attente annule l'unité d'oeuvre.
- Si l'appel fonctionne en dehors d'une unité d'oeuvre définie par l'utilisateur, mais qu'une unité d'oeuvre définie par l'utilisateur existe, le gestionnaire de files d'attente ne peut pas être réassemblé. Si le message ne nécessite pas de réassemblage, l'appel peut tout de même aboutir. Mais si le message nécessite un réassemblage, l'appel échoue avec le code anomalie MQRC\_UOW\_NOT\_AVAILABLE.

Pour les messages non persistants, le gestionnaire de files d'attente ne requiert pas qu'une unité d'oeuvre soit disponible pour effectuer le réassemblage.

Chaque message physique qui est un segment possède son propre descripteur de message. Pour les segments constituant un message logique unique, la plupart des zones du descripteur de message sont identiques pour tous les segments du message logique ; en règle générale, seules les zones `MsgId`, `Offset` et `MsgFlags` diffèrent entre les segments du message logique. Toutefois, si un segment est placé dans une file d'attente de rebut au niveau d'un gestionnaire de files d'attente intermédiaire, le gestionnaire DLQ extrait le message en spécifiant l'option MQGMO\_CONVERT , ce qui peut entraîner la modification du jeu de caractères ou du codage du segment. Si le gestionnaire de la file d'attente des messages non livrés envoie le segment en cours de route, le segment peut avoir un jeu de caractères ou un codage différent des autres segments du message logique lorsque le segment arrive sur le gestionnaire de files d'attente de destination.

Un message logique composé de segments dans lesquels les zones `CodedCharSetId` et `Encoding` diffèrent ne peut pas être réassemblé par le gestionnaire de files d'attente en un seul message logique. A la place, le gestionnaire de files d'attente réassemble et renvoie les premiers segments consécutifs au début du message logique ayant les mêmes identificateurs et codages de jeu de caractères, et l'appel MQGET se termine avec le code achèvement MQCC\_WARNING et le code anomalie MQRC\_INCONSISTENT\_CCIDS ou MQRC\_INCONSISTENT\_ENCODINGS, selon le cas. Cela se produit que MQGMO\_CONVERT soit spécifié ou non. Pour extraire les segments restants, l'application doit réémettre l'appel MQGET sans l'option MQGMO\_COMPLETE\_MSG , en extrayant les segments un par un. MQGMO\_LOGICAL\_ORDER peut être utilisé pour extraire les segments restants dans l'ordre.

Une application qui insère des segments peut également définir d'autres zones dans le descripteur de message avec des valeurs qui diffèrent entre les segments. Toutefois, cela ne présente aucun avantage si l'application de réception utilise MQGMO\_COMPLETE\_MSG pour extraire le message logique. Lorsque le gestionnaire de files d'attente réassemble un message logique, il renvoie dans le descripteur de message les valeurs du descripteur de message pour le premier segment ; la seule






exception est la zone `MsgFlags`, que le gestionnaire de files d'attente définit pour indiquer que le message réassemblé est le seul segment.

Si `MQGMO_COMPLETE_MSG` est spécifié pour un message de rapport, le gestionnaire de files d'attente effectue un traitement spécial. Le gestionnaire de files d'attente vérifie la file d'attente pour voir si tous les messages de rapport de ce type relatifs aux différents segments du message logique sont présents dans la file d'attente. Si tel est le cas, ils peuvent être extraits sous la forme d'un message unique en spécifiant `MQGMO_COMPLETE_MSG`. Pour que cela soit possible, les messages de rapport doivent être générés par un gestionnaire de files d'attente ou un agent MCA prenant en charge la segmentation, ou l'application d'origine doit demander au moins 100 octets de données de message (c'est-à-dire que les options `MQRO_*_WITH_DATA` ou `MQRO_*_WITH_FULL_DATA` appropriées doivent être spécifiées). Si la quantité de données d'application est inférieure à la quantité totale pour un segment, les octets manquants sont remplacés par des valeurs nulles dans le message de rapport renvoyé.

Si `MQGMO_COMPLETE_MSG` est spécifié avec `MQGMO_MSG_UNDER_CURSOR` ou `MQGMO_BROWSE_MSG_UNDER_CURSOR`, le curseur de navigation doit être positionné sur un message dont la zone `Offset` dans `MQMD` a la valeur 0. Si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie `MQRC_INVALID_MSG_UNDER_CURSOR`.

`MQGMO_COMPLETE_MSG` implique `MQGMO_ALL_SEGMENTS_AVAILABLE`, qui n'a donc pas besoin d'être spécifié.

`MQGMO_COMPLETE_MSG` peut être spécifié avec l'une des autres options `MQGMO_*` à l'exception de `MQGMO_SYNCPOINT_IF_PERSISTENT` et avec l'une des options `MQMO_*` à l'exception de `MQMO_MATCH_OFFSET`.

-  **z/OS** Sous z/OS, cette option est prise en charge pour les files d'attente privées et partagées, mais la file d'attente doit avoir un type d'index `MQIT_GROUP_ID`. Pour les files d'attente partagées, l'objet `CFSTRUCT` que la mappe de files d'attente doit être au niveau `CFLEVEL (3)` ou supérieur.
- Sur les plateformes suivantes:
  -  **AIX** AIX
  -  **IBM i** IBM i
  -  **Linux** Linux
  -  **Windows** Windows

et pour IBM MQ MQI clients connecté à ces systèmes, cette option est prise en charge pour toutes les files d'attente locales.

### **MQGMO\_ALL\_MSGS\_AVAILABLE**

Les messages d'un groupe ne peuvent être extraits que lorsque tous les messages du groupe sont disponibles. Si la file d'attente contient des groupes de messages dont certains des messages sont manquants (peut-être parce qu'ils ont été retardés sur le réseau et qu'ils ne sont pas encore arrivés), la spécification de `MQGMO_ALL_MSGS_AVAILABLE` empêche l'extraction des messages appartenant à des groupes incomplets. Toutefois, ces messages contribuent toujours à la valeur de l'attribut de file d'attente **CurrentQDepth**; cela signifie qu'il peut n'y avoir aucun groupe de messages pouvant être extrait, même si `CurrentQDepth` est supérieur à zéro. Si aucun autre message n'est récupérable, le code raison `MQRC_NO_MSG_AVAILABLE` est renvoyé après l'expiration de l'intervalle d'attente indiqué (le cas échéant).

Le traitement de `MQGMO_ALL_MSGS_AVAILABLE` varie selon que `MQGMO_LOGICAL_ORDER` est également spécifié ou non:

- Si les deux options sont spécifiées, `MQGMO_ALL_MSGS_AVAILABLE` n'a d'effet que lorsqu'il n'y a pas de groupe en cours ou de message logique. S'il existe un groupe ou un message logique en cours, `MQGMO_ALL_MSGS_AVAILABLE` est ignoré. Cela signifie que `MQGMO_ALL_MSGS_AVAILABLE` peut rester en fonction lors du traitement des messages dans l'ordre logique.


- Si MQGMO\_ALL\_MSGS\_AVAILABLE est spécifié sans MQGMO\_LOGICAL\_ORDER, MQGMO\_ALL\_MSGS\_AVAILABLE a toujours un effet. Cela signifie que l'option doit être désactivée une fois que le premier message du groupe a été supprimé de la file d'attente, afin de pouvoir supprimer les messages restants du groupe.

L'exécution réussie d'un appel MQGET spécifiant MQGMO\_ALL\_MSGS\_AVAILABLE signifie qu'au moment de l'émission de l'appel MQGET, tous les messages du groupe se trouvaient dans la file d'attente. Toutefois, sachez que d'autres applications peuvent toujours supprimer des messages du groupe (le groupe n'est pas verrouillé pour l'application qui extrait le premier message du groupe).

Si vous omettez cette option, les messages appartenant à des groupes peuvent être extraits même lorsque le groupe est incomplet.

MQGMO\_ALL\_MSGS\_AVAILABLE implique MQGMO\_ALL\_SEGMENTS\_AVAILABLE, qui n'a donc pas besoin d'être spécifié.

MQGMO\_ALL\_MSGS\_AVAILABLE peut être spécifié avec l'une des autres options MQGMO\_\* et avec l'une des options MQMO\_\*.

-  Sous z/OS, cette option est prise en charge pour les files d'attente privées et partagées, mais la file d'attente doit avoir un type d'index MQIT\_GROUP\_ID. Pour les files d'attente partagées, l'objet CFSTRUCT que la mappe de files d'attente doit être au niveau CFLEVEL (3) ou supérieur.

- Sur les plateformes suivantes:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour IBM MQ MQI clients connecté à ces systèmes, cette option est prise en charge pour toutes les files d'attente locales.

### MQGMO\_ALL\_SEGMENTS\_AVAILABLE

Les segments d'un message logique ne peuvent être extraits que lorsque tous les segments du message logique sont disponibles. Si la file d'attente contient des messages segmentés dont certains des segments sont manquants (peut-être parce qu'ils ont été retardés dans le réseau et qu'ils ne sont pas encore arrivés), la spécification de MQGMO\_ALL\_SEGMENTS\_AVAILABLE empêche l'extraction de segments appartenant à des messages logiques incomplets. Toutefois, ces segments contribuent toujours à la valeur de l'attribut de file d'attente **CurrentQDepth**; cela signifie qu'il peut n'y avoir aucun message logique récupérable, même si CurrentQDepth est supérieur à zéro. Si aucun autre message n'est récupérable, le code raison MQRC\_NO\_MSG\_AVAILABLE est renvoyé après l'expiration de l'intervalle d'attente indiqué (le cas échéant).

Le traitement de MQGMO\_ALL\_SEGMENTS\_AVAILABLE varie selon que MQGMO\_LOGICAL\_ORDER est également spécifié ou non:

- Si les deux options sont spécifiées, MQGMO\_ALL\_SEGMENTS\_AVAILABLE n'a d'effet que lorsqu'il n'y a pas de message logique en cours. S'il existe un message logique en cours, MQGMO\_ALL\_SEGMENTS\_AVAILABLE est ignoré. Cela signifie que MQGMO\_ALL\_SEGMENTS\_AVAILABLE peut rester en fonction lors du traitement des messages dans l'ordre logique.
- Si MQGMO\_ALL\_SEGMENTS\_AVAILABLE est spécifié sans MQGMO\_LOGICAL\_ORDER, MQGMO\_ALL\_SEGMENTS\_AVAILABLE a toujours un effet. Cela signifie que l'option doit être désactivée une fois que le premier segment du message logique a été supprimé de la file d'attente, afin de pouvoir supprimer les segments restants du message logique.

Si cette option n'est pas spécifiée, les segments de message peuvent être extraits même lorsque le message logique est incomplet.

Alors que MQGMO\_COMPLETE\_MSG et MQGMO\_ALL\_SEGMENTS\_AVAILABLE exigent que tous les segments soient disponibles avant que l'un d'eux puisse être extrait, le premier renvoie le message complet, tandis que le second permet d'extraire les segments un par un.

Si MQGMO\_ALL\_SEGMENTS\_AVAILABLE est spécifié pour un message de rapport, le gestionnaire de files d'attente vérifie la file d'attente pour voir s'il existe au moins un message de rapport pour chacun des segments qui constituent le message logique complet. Si tel est le cas, la condition MQGMO\_ALL\_SEGMENTS\_AVAILABLE est satisfaite. Toutefois, le gestionnaire de files d'attente ne vérifie pas le *type* des messages de rapport présents, de sorte qu'il peut y avoir un mélange de types de rapport dans les messages de rapport relatifs aux segments du message logique. Par conséquent, la réussite de MQGMO\_ALL\_SEGMENTS\_AVAILABLE n'implique pas que MQGMO\_COMPLETE\_MSG réussira. S'il existe une combinaison de types de rapport pour les segments d'un message logique particulier, ces messages de rapport doivent être extraits un par un.

Vous pouvez spécifier MQGMO\_ALL\_SEGMENTS\_AVAILABLE avec l'une des autres options MQGMO\_\* et avec l'une des options MQMO\_\*.

- Sous z/OS, cette option est prise en charge pour les files d'attente privées et partagées, mais la file d'attente doit avoir un type d'index MQIT\_GROUP\_ID. Pour les files d'attente partagées, l'objet CFSTRUCT que la mappe de files d'attente doit être au niveau CFLEVEL (3) ou supérieur.
- Sur les plateformes suivantes:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour IBM MQ MQI clients connecté à ces systèmes, cette option est prise en charge pour toutes les files d'attente locales.

## Options de propriété

Les options suivantes sont liées aux propriétés du message :

### MQGMO\_PROPERTIES\_AS\_Q\_DEF

Les propriétés du message, à l'exception de celles contenues dans le descripteur de message (ou l'extension), doivent être représentées comme défini par l'attribut de file d'attente **PropertyControl**. Si un `MsgHandle` est fourni, cette option est ignorée et les propriétés du message sont disponibles via `MsgHandle`, sauf si la valeur de l'attribut de file d'attente **PropertyControl** est `MQPROP_FORCE_MQRFH2`.

Il s'agit de l'action par défaut si aucune option de propriétés n'est spécifiée.

### MQGMO\_PROPERTIES\_IN\_HANDLE

Les propriétés du message doivent être mises à disposition via le `MsgHandle`. Si aucun descripteur de message n'est fourni, l'appel échoue avec le code anomalie `MQRC_HMSG_ERROR`.

**Remarque :** Si le message est lu ultérieurement par une application qui ne crée pas de descripteur de message, le gestionnaire de files d'attente place les propriétés de message dans une structure `MQRFH2`. Vous pouvez constater que la présence d'un en-tête `MQRFH2` inattendu perturbe le comportement d'une application existante.

### MQGMO\_NO\_PROPERTIES

Aucune propriété du message, à l'exception de celles contenues dans le descripteur de message (ou l'extension), ne sera extraite. Si un `MsgHandle` est fourni, il est ignoré.

## **MQGMO\_PROPERTIES\_FORCE\_MQRFH2**

Les propriétés du message, à l'exception de celles contenues dans le descripteur de message (ou l'extension), doivent être représentées à l'aide d'en-têtes MQRFH2 . Cela permet d'assurer la compatibilité avec une version antérieure pour les applications qui s'attendent à extraire des propriétés mais qui ne peuvent pas être modifiées pour utiliser des descripteurs de message. Si un `MsgHandle` est fourni, il est ignoré.

## **MQGMO\_PROPERTIES\_COMPATIBILITY**

Si le message contient une propriété avec le préfixe "**mcd.**", "**jms.**", "**usr.**" ou "**mqext.**", toutes les propriétés de message sont distribuées à l'application dans un en-tête MQRFH2 . Sinon, toutes les propriétés du message, à l'exception de celles du descripteur de message (ou extension), sont supprimées et ne sont plus accessibles à l'application.

## **Option par défaut**

Si aucune des options décrites n'est requise, l'option suivante peut être utilisée:

### **MQGMO\_NONE**

Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. Toutes les options sont définies sur leur valeur par défaut. MQGMO\_NONE aide la documentation du programme ; il n'est pas prévu que cette option soit utilisée avec d'autres, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

La valeur initiale de la zone Options est MQGMO\_NO\_WAIT plus MQGMO\_PROPERTIES\_AS\_Q\_DEF.

## ***WaitInterval (MQLONG) pour MQGMO***

Il s'agit de la durée approximative, exprimée en millisecondes, pendant laquelle l'appel MQGET attend l'arrivée d'un message approprié (c'est-à-dire un message satisfaisant aux critères de sélection spécifiés dans le paramètre **MsgDesc** de l'appel MQGET).

**Important :** Il n'y a pas d'attente, ni de délai, si un message approprié est disponible immédiatement.

Voir la zone *MsgId* décrite dans «MQMD-Descripteur de message», à la page 438 pour plus de détails). Si aucun message approprié n'est arrivé après ce délai, l'appel se termine avec MQCC\_FAILED et le code anomalie MQRC\_NO\_MSG\_AVAILABLE.

Sous z/OS, la période pendant laquelle l'appel MQGET attend est affectée par les considérations relatives au chargement et à la planification des travaux du système et peut varier entre la valeur spécifiée pour *WaitInterval* et une valeur supérieure d'environ 100 millisecondes à *WaitInterval*.

*WaitInterval* est utilisé avec l'option MQGMO\_WAIT ou MQGMO\_SET\_SIGNAL. Elle est ignorée si aucune de ces deux options n'est spécifiée. Si l'une de ces valeurs est indiquée, *WaitInterval* doit être supérieur ou égal à zéro, ou la valeur spéciale suivante:

### **MQWI\_ILLIMITÉ**

Intervalle d'attente illimité.

La valeur initiale de cette zone est 0.

## ***Signal1 (MQLONG) pour MQGMO***

Il s'agit d'une zone d'entrée qui est utilisée uniquement avec l'option MQGMO\_SET\_SIGNAL ; elle identifie un signal à distribuer lorsqu'un message est disponible.

**Remarque :** Le type de données et l'utilisation de cette zone sont déterminés par l'environnement ; pour cette raison, les applications que vous souhaitez porter entre différents environnements ne doivent pas utiliser de signaux.

- Sous z/OS, cette zone doit contenir l'adresse d'un bloc de contrôle d'événement (ECB). L'ECB doit être effacé par l'application avant que l'appel MQGET ne soit émis. La mémoire contenant le ECB ne doit pas être libérée tant que la file d'attente n'est pas fermée. La commande ECB est envoyée par le gestionnaire de files d'attente avec l'un des codes d'achèvement de signal décrits. Ces codes



achèvement sont définis dans les bits 2 à 31 de l'ECB, la zone définie dans la macro de mappage z/OS IHAECB pour un code achèvement utilisateur.

- Dans tous les autres environnements, il s'agit d'une zone réservée ; sa valeur n'est pas significative.

Les codes achèvement du signal sont les suivants:

#### **MQEC\_MSG\_ARRIVÉ**

Un message approprié est arrivé dans la file d'attente. Ce message n'a pas été réservé à l'appelant ; une deuxième demande MQGET doit être émise, mais une autre application peut extraire le message avant que la deuxième demande ne soit effectuée.

#### **MQEC\_INTERVALLE\_ATTENTE\_EXPIRÉE**

Le *WaitInterval* spécifié a expiré sans qu'un message approprié n'arrive.

#### **MQEC\_ATTENTE\_ANNULÉ**

L'attente a été annulée pour une raison indéterminée (telle que l'arrêt du gestionnaire de files d'attente ou la désactivation de la file d'attente). Emettez à nouveau la demande si vous souhaitez un diagnostic plus approfondi.

#### **MQEC\_Q\_MGR\_QUIESCING**

L'attente a été annulée car le gestionnaire de files d'attente est passé à l'état de mise au repos (MQGMO\_FAIL\_IF\_QUIESCING a été spécifié sur l'appel MQGET).

#### **MQEC\_CONNEXION\_MISE au repos**

L'attente a été annulée car la connexion est entrée dans l'état de mise au repos (MQGMO\_FAIL\_IF\_QUIESCING a été spécifié dans l'appel MQGET).

La valeur initiale de cette zone est déterminée par l'environnement:

- Sous z/OS, la valeur initiale est le pointeur null.
- Dans tous les autres environnements, la valeur initiale est 0.

### ***Signal2 (MQLONG) pour MQGMO***

Il s'agit d'une zone d'entrée utilisée uniquement avec l'option MQGMO\_SET\_SIGNAL. Il s'agit d'une zone réservée ; sa valeur n'est pas significative.

La valeur initiale de cette zone est 0.

### ***ResolvedQName (MQCHAR48) pour MQGMO***

Il s'agit d'une zone de sortie que le gestionnaire de files d'attente définit sur le nom local de la file d'attente à partir de laquelle le message a été extrait, comme défini pour le gestionnaire de files d'attente local. Ce nom est différent du nom utilisé pour ouvrir la file d'attente si:

- Une file d'attente alias a été ouverte (auquel cas, le nom de la file d'attente locale dans laquelle l'alias a été résolu est renvoyé), ou
- Une file d'attente modèle a été ouverte (auquel cas, le nom de la file d'attente locale dynamique est renvoyé).

La longueur de cette zone est indiquée par MQ\_Q\_NAME\_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

### ***MatchOptions (MQLONG) pour MQGMO***

Ces options permettent à l'application de choisir les zones du paramètre **MsgDesc** à utiliser pour sélectionner le message renvoyé par l'appel MQGET. L'application définit les options requises dans cette zone, puis définit les zones correspondantes dans le paramètre **MsgDesc** sur les valeurs requises pour ces zones. Seuls les messages ayant ces valeurs dans le MQMD pour le message sont candidats à l'extraction à l'aide de ce paramètre **MsgDesc** dans l'appel MQGET. Les zones pour lesquelles l'option de correspondance correspondante n'est pas spécifiée sont ignorées lors de la sélection du message à renvoyer. Si vous n'indiquez aucun critère de sélection dans l'appel MQGET (c'est-à-dire *tout message* est acceptable), définissez *MatchOptions* sur MQMO\_NONE.

- Sous z/OS, les critères de sélection pouvant être utilisés peuvent être limités par le type d'index utilisé pour la file d'attente. Pour plus de détails, voir l'attribut de file d'attente **IndexType**.

Si vous spécifiez MQGMO\_LOGICAL\_ORDER, seuls certains messages peuvent être renvoyés par l'appel MQGET suivant:

- S'il n'y a pas de groupe ou de message logique en cours, seuls les messages dont la valeur est *MsgSeqNumber* égale à 1 et *Offset* égale à 0 peuvent être renvoyés. Dans ce cas, vous pouvez utiliser une ou plusieurs des options de correspondance suivantes pour sélectionner les messages éligibles renvoyés:
  - ID MQMO\_MATCH\_MSG\_ID
  - ID\_CORREL\_MQMO\_MATCH\_
  - ID\_GROUPE\_MQMO\_MATCH
- S'il existe un groupe ou un message logique en cours, seul le message suivant du groupe ou du segment suivant du message logique peut être renvoyé, ce qui ne peut pas être modifié en spécifiant les options MQMO\_\*

Dans les deux cas précédents, vous pouvez spécifier des options de correspondance qui ne s'appliquent pas, mais la valeur de la zone appropriée dans le paramètre **MsgDesc** doit correspondre à la valeur de la zone correspondante dans le message à renvoyer. L'appel échoue avec le code anomalie MQRC\_MATCH\_OPTIONS\_ERROR si cette condition n'est pas remplie.

*MatchOptions* est ignoré si vous spécifiez MQGMO\_MSG\_UNDER\_CURSOR ou MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR.

L'obtention de messages basés sur une propriété de message n'est pas effectuée à l'aide des options de correspondance ; pour plus d'informations, voir «[SelectionString \(MQCHARV\) pour MQOD](#)», à la page 515.

Vous pouvez spécifier une ou plusieurs des options de correspondance suivantes:

#### **ID MQMO\_MATCH\_MSG\_ID**

Le message à extraire doit avoir un identificateur de message qui correspond à la valeur de la zone *MsgId* dans le paramètre **MsgDesc** de l'appel MQGET. Cette correspondance s'ajoute à toute autre correspondance pouvant s'appliquer (par exemple, l'identificateur de corrélation).

Si vous omettez cette option, la zone *MsgId* du paramètre **MsgDesc** est ignorée et tout identificateur de message correspond.

**Remarque :** L'identificateur de message MQMI\_NONE est une valeur spéciale qui correspond à n'importe quel identificateur de message dans le MQMD pour le message. Par conséquent, la spécification de MQMO\_MATCH\_MSG\_ID avec MQMI\_NONE est identique à la spécification de MQMO\_MATCH\_MSG\_ID.

#### **ID\_CORREL\_MQMO\_MATCH\_**

Le message à extraire doit avoir un identificateur de corrélation qui correspond à la valeur de la zone *CorrelId* dans le paramètre **MsgDesc** de l'appel MQGET. Cette correspondance s'ajoute à toute autre correspondance pouvant s'appliquer (par exemple, l'identificateur de message).

Si vous omettez cette option, la zone *CorrelId* du paramètre **MsgDesc** est ignorée et tout identificateur de corrélation correspond.

**Remarque :** L'identificateur de corrélation MQCI\_NONE est une valeur spéciale qui correspond à *tout* identificateur de corrélation dans le MQMD du message. Par conséquent, la spécification de MQMO\_MATCH\_CORREL\_ID avec MQCI\_NONE est identique à la spécification de MQMO\_MATCH\_CORREL\_ID.

#### **ID\_GROUPE\_MQMO\_MATCH**

Le message à extraire doit avoir un identificateur de groupe qui correspond à la valeur de la zone *GroupId* dans le paramètre **MsgDesc** de l'appel MQGET. Cette correspondance s'ajoute à toute autre correspondance pouvant s'appliquer (par exemple, l'identificateur de corrélation).

Si vous omettez cette option, la zone *GroupId* du paramètre **MsgDesc** est ignorée et tout identificateur de groupe correspond.

**Remarque :** L'identificateur de groupe MQGI\_NONE est une valeur spéciale qui correspond à *n'importe quel* identificateur de groupe dans le MQMD pour le message. Par conséquent, la spécification de MQMO\_MATCH\_GROUP\_ID avec MQGI\_NONE est identique à la spécification de MQMO\_MATCH\_GROUP\_ID.

#### **NUMERO MQMO\_MATCH\_MSG\_SEQ\_NO**

Le message à extraire doit avoir un numéro de séquence de message qui correspond à la valeur de la zone *MsgSeqNumber* dans le paramètre **MsgDesc** de l'appel MQGET. Cette correspondance s'ajoute à toute autre correspondance pouvant s'appliquer (par exemple, l'identificateur de groupe).

Si vous omettez cette option, la zone *MsgSeqNumber* du paramètre **MsgDesc** est ignorée et tout numéro de séquence de message correspond.

#### **MQMO\_MATCH\_OFFSET**

Le message à extraire doit avoir un décalage qui correspond à la valeur de la zone *Offset* dans le paramètre **MsgDesc** de l'appel MQGET. Cette correspondance s'ajoute à toute autre correspondance pouvant s'appliquer (par exemple, le numéro de séquence du message).

Si vous omettez cette option, la zone *Offset* du paramètre **MsgDesc** est ignorée et tout décalage correspond.

- Cette option n'est pas prise en charge sous z/OS.

#### **MQMO\_MATCH\_MSG\_TOKEN**

Le message à extraire doit avoir un jeton de message qui correspond à la valeur de la zone *MsgToken* dans la structure MQGMO définie dans l'appel MQGET.

Vous pouvez spécifier cette option pour toutes les files d'attente locales. Si vous le spécifiez pour une file d'attente dont le *IndexType* est MQIT\_MSG\_TOKEN (file d'attente gérée par WLM), vous ne pouvez spécifier aucune autre option de correspondance avec MQMO\_MATCH\_MSG\_TOKEN.

Vous ne pouvez pas spécifier MQMO\_MATCH\_MSG\_TOKEN avec MQGMO\_WAIT ou MQGMO\_SET\_SIGNAL. Si l'application souhaite attendre qu'un message arrive dans une file d'attente dont le *IndexType* est MQIT\_MSG\_TOKEN, indiquez MQMO\_NONE.

Si vous omettez cette option, la zone *MsgToken* de MQGMO est ignorée et tout jeton de message correspond.

Si vous n'indiquez aucune des options décrites, vous pouvez utiliser l'option suivante:

#### **MQMO\_AUCUN**

N'utilisez aucune correspondance pour sélectionner le message à renvoyer ; tous les messages de la file d'attente peuvent être extraits (mais soumis au contrôle des options MQGMO\_ALL\_MSGS\_AVAILABLE, MQGMO\_ALL\_SEGMENTS\_AVAILABLE et MQGMO\_COMPLETE\_MSG).

MQMO\_NONE facilite la documentation du programme. Il n'est pas prévu que cette option soit utilisée avec une autre option MQMO\_\*, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est MQMO\_MATCH\_MSG\_ID avec MQMO\_MATCH\_CORREL\_ID. Cette zone est ignorée si *Version* est inférieur à MQGMO\_VERSION\_2.

**Remarque :** La valeur initiale de la zone *MatchOptions* est définie à des fins de compatibilité avec les gestionnaires de files d'attente MQSeries antérieurs. Toutefois, lors de la lecture d'une série de messages à partir d'une file d'attente sans utiliser de critères de sélection, cette valeur initiale nécessite que l'application réinitialise les zones *MsgId* et *CorrelId* sur MQMI\_NONE et MQCI\_NONE avant chaque appel MQGET. Evitez de réinitialiser *MsgId* et *CorrelId* en définissant *Version* sur MQGMO\_VERSION\_2 et *MatchOptions* sur MQMO\_NONE.

#### **Concepts associés**

Sélecteurs de message dans JMS

### **GroupStatus (MQCHAR) pour MQGMO**

Cet indicateur indique si le message extrait se trouve dans un groupe.

Elle a l'une des valeurs suivantes :

#### **MQGS\_NOT\_IN\_GROUPE**

Le message n'est pas dans un groupe.

#### **MQGS\_MSG\_IN\_GROUPE**

Le message se trouve dans un groupe, mais n'est pas le dernier du groupe.

#### **MQGS\_LAST\_MSG\_IN\_GROUP**

Le message est le dernier du groupe.

Il s'agit également de la valeur renvoyée si le groupe se compose d'un seul message.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est MQGS\_NOT\_IN\_GROUP. Cette zone est ignorée si *Version* est inférieur à MQGMO\_VERSION\_2.

### **SegmentStatus (MQCHAR) pour MQGMO**

Il s'agit d'un indicateur qui indique si le message extrait est un segment d'un message logique. Elle a l'une des valeurs suivantes :

#### **MQSS\_NON\_SEGMENT**

Le message n'est pas un segment.

#### **SEGMENT\_MQSS\_SEGMENT**

Le message est un segment, mais n'est pas le dernier segment du message logique.

#### **SEGMENT\_LAST\_MQSS\_**

Le message est le dernier segment du message logique.

Il s'agit également de la valeur renvoyée si le message logique se compose d'un seul segment.

Sous z/OS, le gestionnaire de files d'attente définit toujours cette zone sur MQSS\_NOT\_A\_SEGMENT.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est MQSS\_NOT\_A\_SEGMENT. Cette zone est ignorée si *Version* est inférieur à MQGMO\_VERSION\_2.

### **Segmentation (MQCHAR) pour MQGMO**

Il s'agit d'un indicateur qui indique si une segmentation supplémentaire est autorisée pour le message extrait. Elle a l'une des valeurs suivantes :

#### **MQSEG\_INHIBÉ**

Segmentation non autorisée.

#### **MQSEG\_AUTORISÉ\_**

Segmentation autorisée.

Sous z/OS, le gestionnaire de files d'attente définit toujours cette zone sur MQSEG\_INHIBÉ.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est MQSEG\_INHIBÉE. Cette zone est ignorée si *Version* est inférieur à MQGMO\_VERSION\_2.

### **Reserved1 (MQCHAR) pour MQGMO**

Il s'agit d'une zone réservée. La valeur initiale de cette zone est un caractère blanc. Cette zone est ignorée si *Version* est inférieur à MQGMO\_VERSION\_2.

### **MsgToken (MQBYTE16) pour MQGMO**

Zone MsgToken -Structure MQGMO. Cette zone est utilisée par le gestionnaire de files d'attente pour identifier un message de manière unique.

Il s'agit d'une chaîne d'octets générée par le gestionnaire de files d'attente pour identifier un message de manière unique dans une file d'attente. Le jeton de message est généré lorsque le message est placé pour la première fois sur le gestionnaire de files d'attente et reste avec le message jusqu'à ce qu'il soit

définitivement supprimé du gestionnaire de files d'attente, sauf si le gestionnaire de files d'attente est redémarré.

Lorsque le message est supprimé de la file d'attente, le *MsgToken* qui a identifié cette instance du message n'est plus valide et n'est jamais réutilisé. Si le gestionnaire de files d'attente est redémarré, le *MsgToken* qui a identifié un message dans la file d'attente avant le redémarrage peut ne pas être valide après le redémarrage. Toutefois, *MsgToken* n'est jamais réutilisé pour identifier une autre instance de message. *MsgToken* est généré par le gestionnaire de files d'attente et n'est visible par aucune application externe.

Lorsqu'un message est renvoyé par un appel à MQGET dans lequel un MQGMO version 3 ou ultérieure est fourni, le *MsgToken* identifiant le message dans la file d'attente est renvoyé dans le MQGMO par le gestionnaire de files d'attente. Il existe une exception: lorsque le message est supprimé de la file d'attente en dehors du point de synchronisation, le gestionnaire de files d'attente peut ne pas renvoyer de *MsgToken* car il n'est pas utile d'identifier le message renvoyé lors d'un appel MQGET ultérieur. Les applications doivent uniquement utiliser *MsgToken* pour faire référence au message lors des appels MQGET suivants.

Si un *MsgToken* est fourni et que *MatchOption* MQMO\_MATCH\_MSG\_TOKEN est spécifié et que ni MQGMO\_MSG\_UNDER\_CURSOR ni MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR ne sont spécifiés, seul le message identifié par ce *MsgToken* peut être renvoyé. L'option est valide sur toutes les files d'attente locales, indépendamment de INDXTYPE, et sur z/OS, vous devez utiliser INDXTYPE (MSGTOKEN) uniquement sur les files d'attente Workload Manager (WLM).

Tous les autres *MatchOptions* spécifiés sont vérifiés et, s'ils ne correspondent pas, MQRC\_NO\_MSG\_AVAILABLE est renvoyé. Si MQGMO\_BROWSE\_NEXT est codé avec MQMO\_MATCH\_MSG\_TOKEN, le message identifié par *MsgToken* est renvoyé uniquement s'il dépasse le curseur de navigation pour le descripteur appelant.

Si MQGMO\_MSG\_UNDER\_CURSOR ou MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR est spécifié, MQMO\_MATCH\_MSG\_TOKEN est ignoré.

MQMO\_MATCH\_MSG\_TOKEN n'est pas valide avec les options d'obtention de message suivantes:

- MQGMO\_WAIT
- MQGMO\_SET\_SIGNAL

Pour un appel MQGET spécifiant MQMO\_MATCH\_MSG\_TOKEN, un MQGMO de version 3 ou ultérieure doit être fourni à l'appel, sinon MQRC\_WRONG\_GMO\_VERSION est renvoyé.

Si *MsgToken* n'est pas valide pour le moment, MQCC\_FAILED avec MQRC\_NO\_MSG\_AVAILABLE est renvoyé, sauf s'il existe une autre erreur.

### ***ReturnedLength (MQLONG) pour MQGMO***

Il s'agit d'une zone de sortie que le gestionnaire de files d'attente définit sur la longueur en octets des données de message renvoyées par l'appel MQGET dans le paramètre **Buffer**. Si le gestionnaire de files d'attente ne prend pas en charge cette fonction, *ReturnedLength* est défini sur la valeur MQRL\_UNDEFINED.

Lorsque des messages sont convertis entre des codages ou des jeux de caractères, les données de message peuvent parfois changer de taille. Au retour de l'appel MQGET:

- Si *ReturnedLength* n'est pas MQRL\_UNDEFINED, le nombre d'octets de données de message renvoyés est indiqué par *ReturnedLength*.
- Si *ReturnedLength* a la valeur MQRL\_UNDEFINED, le nombre d'octets de données de message renvoyés est généralement indiqué par la plus petite des valeurs *BufferLength* et *DataLength*, mais peut être *inférieur* à si l'appel MQGET se termine avec le code anomalie MQRC\_TRUNCATED\_MSG\_ACCEPTED. Dans ce cas, les octets non significatifs du paramètre **Buffer** sont définis sur des valeurs nulles.

La valeur spéciale suivante est définie:

## **MQRL\_UNDEFINED**

Longueur des données renvoyées non définie.

Sous z/OS, la valeur renvoyée pour la zone *ReturnedLength* est toujours MQRL\_UNDEFINED.

La valeur initiale de cette zone est MQRL\_UNDEFINED. Cette zone est ignorée si *Version* est inférieur à MQGMO\_VERSION\_3.

## **Reserved2 (MQLONG) pour MQGMO**

Il s'agit d'une zone réservée. La valeur initiale de cette zone est un caractère blanc. Cette zone est ignorée si *Version* est inférieur à MQGMO\_VERSION\_4.

## **MsgHandle (MQHMSG) pour MQGMO**

Si l'option MQGMO\_PROPERTIES\_AS\_Q\_DEF est spécifiée et que l'attribut de file d'attente **PropertyControl** n'est pas défini sur MQPROP\_FORCE\_MQRFH2, il s'agit du descripteur d'un message qui sera rempli avec les propriétés du message extrait de la file d'attente. Le descripteur est créé par un appel MQCRTMH. Toutes les propriétés déjà associées à l'identificateur seront effacées avant l'extraction d'un message.

La valeur suivante peut également être spécifiée:

MQHM\_NONE

Aucun descripteur de message fourni.

Aucun descripteur de message n'est requis dans l'appel MQGET si un descripteur de message valide est fourni et utilisé dans la sortie pour contenir les propriétés de message, le descripteur de message associé au descripteur de message est utilisé pour les zones d'entrée.

Si un descripteur de message est spécifié dans l'appel MQGET, il est toujours prioritaire sur le descripteur de message associé à un descripteur de message.

Si MQGMO\_PROPERTIES\_FORCE\_MQRFH2 est spécifié ou que MQGMO\_PROPERTIES\_AS\_Q\_DEF est spécifié et que l'attribut de file d'attente **PropertyControl** est MQPROP\_FORCE\_MQRFH2, l'appel échoue avec le code anomalie MQRC\_MD\_ERROR lorsqu'aucun paramètre de descripteur de message n'est spécifié.

Lors du retour de l'appel MQGET, les propriétés et le descripteur de message associés à cet identificateur de message sont mis à jour pour refléter l'état du message extrait (ainsi que le descripteur de message s'il a été fourni sur l'appel MQGET). Les propriétés du message peuvent ensuite être recherchées à l'aide de l'appel MQINQMP.

A l'exception des extensions de descripteur de message, lorsqu'elles sont présentes, une propriété qui peut être interrogée avec l'appel MQINQMP n'est pas contenue dans les données de message ; si le message dans la file d'attente contient des propriétés dans les données de message, elles sont supprimées des données de message avant que les données ne soient renvoyées à l'application.

Si aucun descripteur de message n'est fourni ou si la version est inférieure à MQGMO\_VERSION\_4, vous devez fournir un descripteur de message valide dans l'appel MQGET. Toutes les propriétés de message (à l'exception de celles contenues dans le descripteur de message) sont renvoyées dans le sujet de données de message avec la valeur des options de propriété dans la structure MQGMO et l'attribut de file d'attente **PropertyControl**.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQHM\_NONE. Cette zone est ignorée si **Version** est inférieur à MQGMO\_VERSION\_4.

## **MQIIH - En-tête d'informations IMS**

La structure MQIIH décrit les informations d'en-tête d'un message envoyé à IMS via le pont IMS. Pour toute plateforme prise en charge par IBM MQ, vous pouvez créer et transmettre un message qui inclut

la structure MQIIH, mais seul un gestionnaire de files d'attente IBM MQ for z/OS peut utiliser le pont IMS . Par conséquent, pour que le message puisse être envoyé à IMS à partir d'un gestionnaire de files d'attente nonz/OS , votre réseau de gestionnaires de files d'attente doit inclure au moins un gestionnaire de files d'attente z/OS par lequel le message peut être acheminé.

## Disponibilité

Tous les systèmes IBM MQ et clients IBM MQ .

## Format

MQFMT\_IMS

## Jeu de caractères et codage

Des conditions spéciales s'appliquent au jeu de caractères et au codage utilisés pour la structure MQIIH et les données de message d'application:

- Les applications qui se connectent au gestionnaire de files d'attente qui possède la file d'attente de pont IMS doivent fournir une structure MQIIH qui se trouve dans le jeu de caractères et le codage du gestionnaire de files d'attente. En effet, la conversion de données de la structure MQIIH n'est pas effectuée dans ce cas.
- Les applications qui se connectent à d'autres gestionnaires de files d'attente peuvent fournir une structure MQIIH qui se trouve dans l'un des jeux de caractères et codages pris en charge ; l'agent MCA récepteur connecté au gestionnaire de files d'attente propriétaire de la file d'attente de pont IMS convertit le MQIIH.
- Les données de message d'application qui suivent la structure MQIIH doivent être dans le même jeu de caractères et le même codage que la structure MQIIH. N'utilisez pas les zones *CodedCharSetId* et *Encoding* de la structure MQIIH pour spécifier le jeu de caractères et le codage des données de message d'application.

Vous devez fournir un exit de conversion de données pour convertir les données de message d'application si les données ne sont pas l'un des formats intégrés pris en charge par le gestionnaire de files d'attente.

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID MQIIH_STRUC_ID	' IIH¬ '
<u>Version</u> (numéro de version de la structure)	MQIIH_VERSION_1	1
<u>StrucLength</u> (longueur de la structure MQIIH)	MQIIH_LENGTH_1	84
<u>Codage</u> (réservé-voir «Jeu de caractères et codage», à la page 423)	Aucun	0
<u>CodedCharSetId</u> (réservé-voir «Jeu de caractères et codage», à la page 423)	Aucun	0
<u>Format</u> (nom de formatMQ des données qui suivent MQIIH)	MQFMT_AUCUN	Espaces vides

Tableau 497. Zones dans MQIIH pour MQIIH (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
Indicateurs (indicateurs)	MQIIH_NONE	0
LTermOverride (remplacement de terminal logique)	Aucun	Espaces vides
MFSMapName (nom de mappe des services de format de message)	Aucun	Espaces vides
ReplyToReplyTo (nom de formatMQ du message de réponse)	MQFMT_AUCUN	Espaces vides
Authentificateur (mot de passe ou mot de passeRACF)	MQIAUT_AUCUN	Espaces vides
TranInstanceID (identificateur de l'instance de transaction)	MQITII_AUCUN	Valeurs NULL
TranState (état de la transaction)	MQITS_NON_CONVERSATION	'¬'
CommitMode (mode de validation)	MQICM_COMMIT_THEN_SEND	'0'
SecurityScope (portée de sécurité)	CONTROLE MQISS_CHECK	'C'
Réservé (réservé)	Aucun	'¬'
<p><b>Remarques :</b></p> <ol style="list-style-type: none"> <li>1. Le symbole ¬ représente un caractère blanc unique.</li> <li>2. Dans le langage de programmation C, la variable macroMQIIH_DEFAULT contient les valeurs répertoriées dans la table. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</li> </ol> <pre>MQIIH MyIIH = {MQIIH_DEFAULT};</pre>		

## Déclarations de langue

Déclaration C pour MQIIH

```
typedef struct tagMQIIH MQIIH;
struct tagMQIIH {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    StrucLength;      /* Length of MQIIH structure */
    MQLONG    Encoding;         /* Reserved */
    MQLONG    CodedCharSetId;   /* Reserved */
    MQCHAR8   Format;           /* MQ format name of data that follows
                                MQIIH */
    MQLONG    Flags;            /* Flags */
    MQCHAR8   LTermOverride;    /* Logical terminal override */
    MQCHAR8   MFSMapName;       /* Message format services map name */
    MQCHAR8   ReplyToFormat;    /* MQ format name of reply message */
    MQCHAR8   Authenticator;     /* RACF password or passticket */
    MQBYTE16  TranInstanceId;    /* Transaction instance identifier */
    MQCHAR    TranState;        /* Transaction state */
    MQCHAR    CommitMode;       /* Commit mode */
    MQCHAR    SecurityScope;    /* Security scope */
};
```



```

MQCHAR    Reserved;          /* Reserved */
};

```

## Déclaration COBOL pour MQIIH

```

** MQIIH structure
10 MQIIH.
**   Structure identifier
15 MQIIH-STRUCID      PIC X(4).
**   Structure version number
15 MQIIH-VERSION     PIC S9(9) BINARY.
**   Length of MQIIH structure
15 MQIIH-STRUCLNGTH  PIC S9(9) BINARY.
**   Reserved
15 MQIIH-ENCODING    PIC S9(9) BINARY.
**   Reserved
15 MQIIH-CODEDCHARSETID PIC S9(9) BINARY.
**   MQ format name of data that follows MQIIH
15 MQIIH-FORMAT      PIC X(8).
**   Flags
15 MQIIH-FLAGS       PIC S9(9) BINARY.
**   Logical terminal override
15 MQIIH-LTERMOVERRIDE PIC X(8).
**   Message format services map name
15 MQIIH-MFSMAPNAME  PIC X(8).
**   MQ format name of reply message
15 MQIIH-REPLYTOFORMAT PIC X(8).
**   RACF password or passticket
15 MQIIH-AUTHENTICATOR PIC X(8).
**   Transaction instance identifier
15 MQIIH-TRANINSTANCEID PIC X(16).
**   Transaction state
15 MQIIH-TRANSTATE   PIC X.
**   Commit mode
15 MQIIH-COMMITMODE  PIC X.
**   Security scope
15 MQIIH-SECURITYSCOPE PIC X.
**   Reserved
15 MQIIH-RESERVED    PIC X.

```

## Déclaration PL/I pour MQIIH

```

dcl
1 MQIIH based,
3 StrucId      char(4),          /* Structure identifier */
3 Version      fixed bin(31),   /* Structure version number */
3 StrucLength  fixed bin(31),   /* Length of MQIIH structure */
3 Encoding     fixed bin(31),   /* Reserved */
3 CodedCharSetId fixed bin(31), /* Reserved */
3 Format        char(8),         /* MQ format name of data that follows
MQIIH */
3 Flags        fixed bin(31),   /* Flags */
3 LTermOverride char(8),       /* Logical terminal override */
3 MFSMapName   char(8),        /* Message format services map name */
3 ReplyToFormat char(8),       /* MQ format name of reply message */
3 Authenticator char(8),       /* RACF password or passticket */
3 TranInstanceId char(16),    /* Transaction instance identifier */
3 TranState    char(1),        /* Transaction state */
3 CommitMode   char(1),        /* Commit mode */
3 SecurityScope char(1),      /* Security scope */
3 Reserved     char(1);        /* Reserved */

```

## Déclaration High Level Assembler pour MQIIH

MQIIH	DSECT	
MQIIH_STRUCID	DS CL4	Structure identifier
MQIIH_VERSION	DS F	Structure version number
MQIIH_STRUCLNGTH	DS F	Length of MQIIH structure
MQIIH_ENCODING	DS F	Reserved
MQIIH_CODEDCHARSETID	DS F	Reserved
MQIIH_FORMAT	DS CL8	MQ format name of data that follows
*		MQIIH
MQIIH_FLAGS	DS F	Flags
MQIIH_LTERMOVERRIDE	DS CL8	Logical terminal override
MQIIH_MFSMAPNAME	DS CL8	Message format services map name

MQIIH_REPLYTOFORMAT	DS	CL8	MQ format name of reply message
MQIIH_AUTHENTICATOR	DS	CL8	RACF password or passticket
MQIIH_TRANINSTANCEID	DS	XL16	Transaction instance identifier
MQIIH_TRANSTATE	DS	CL1	Transaction state
MQIIH_COMMITMODE	DS	CL1	Commit mode
MQIIH_SECURITYSCOPE	DS	CL1	Security scope
MQIIH_RESERVED	DS	CL1	Reserved
*			
MQIIH_LENGTH	EQU	*-MQIIH	
	ORG	MQIIH	
MQIIH_AREA	DS	CL(MQIIH_LENGTH)	

## Déclaration Visual Basic pour MQIIH

```

Type MQIIH
  StrucId      As String*4 'Structure identifier'
  Version     As Long      'Structure version number'
  StrucLength As Long      'Length of MQIIH structure'
  Encoding    As Long      'Reserved'
  CodedCharSetId As Long  'Reserved'
  Format      As String*8  'MQ format name of data that follows MQIIH'
  Flags      As Long      'Flags'
  LTermOverride As String*8 'Logical terminal override'
  MFSSMapName As String*8  'Message format services map name'
  ReplyToFormat As String*8 'MQ format name of reply message'
  Authenticator As String*8 'RACF password or passticket'
  TranInstanceId As MQBYTE16 'Transaction instance identifier'
  TranState     As String*1 'Transaction state'
  CommitMode   As String*1 'Commit mode'
  SecurityScope As String*1 'Security scope'
  Reserved     As String*1 'Reserved'
End Type

```

### **StrucId (MQCHAR4) pour MQIIH**

Il s'agit de l'identificateur de structure de la structure d'en-tête d'informations IMS . Il s'agit toujours d'une zone d'entrée. Sa valeur est MQIIH\_STRUC\_ID.

La valeur doit être:

#### **ID MQIIH\_STRUC\_ID**

Identificateur de la structure d'en-tête d'informations IMS .

Pour le langage de programmation C, la constante MQIIH\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQIIH\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQIIH**

Il s'agit du numéro de version de la structure. La valeur doit être:

#### **MQIIH\_VERSION\_1**

Numéro de version de la structure d'en-tête d'informations IMS .

La constante suivante indique le numéro de version de la version en cours:

#### **MQIIH\_VERSION\_CURRENT\_**

Version actuelle de la structure d'en-tête d'informations IMS .

La valeur initiale de cette zone est MQIIH\_VERSION\_1.

### **StrucLength (MQLONG) pour MQIIH**

Longueur de la structure MQIIH. La valeur doit être:

#### **MQIIH\_LENGTH\_1**

Longueur de la structure d'en-tête d'informations IMS .

La valeur initiale de cette zone est MQIIH\_LENGTH\_1.

### **Codage (MQLONG) pour MQIIH**

Il s'agit d'une zone réservée ; sa valeur n'est pas significative. La valeur initiale de cette zone est 0.

Le codage des structures prises en charge qui suivent une structure MQIIH est identique à celui de la structure MQIIH elle-même et provient de tout en-tête MQ précédent.

### ***CodedCharSetId (MQLONG) pour MQIIH***

Il s'agit d'une zone réservée ; sa valeur n'est pas significative. La valeur initiale de cette zone est 0.

L'ID de jeu de caractères pour les structures prises en charge qui suivent une structure MQIIH est identique à celui de la structure MQIIH elle-même et provient de tout en-tête MQ précédent.

### ***Format (MQCHAR8) pour MQIIH***

Indique le nom de format MQ des données qui suivent la structure MQIIH.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données.

La longueur de cette zone est indiquée par MQ\_FORMAT\_LENGTH. La valeur initiale de cette zone est MQFMT\_NONE.

### ***Indicateurs (MQLONG) pour MQIIH***

La valeur des indicateurs doit être:

#### **MQIIH\_AUCUN**

Aucun indicateur.

#### **MQIIH\_PASS\_EXPIRATION**

Le message de réponse contient:

- Les mêmes options de rapport d'expiration que le message de demande
- Délai d'expiration restant à partir du message de demande sans ajustement effectué pour le temps de traitement du pont

Si cette valeur n'est pas définie, le délai d'expiration est défini sur *unlimited*.

#### **MQIIH\_REPLY\_FORMAT\_NONE**

Définit MQIIH.Format de la réponse à MQFMT\_NONE.

#### **MQIIH\_IGNORE\_PURG**

Définit l'indicateur TMAMIPRG dans le préfixe OTMA, qui demande à OTMA d'ignorer les appels PURG sur le PCB TP pour les transactions CMO .

#### **MQIIH\_CMO\_REQUEST\_RESPONSE**

Pour les transactions en mode de validation 0 (CMO), cet indicateur définit l'indicateur TMAMHRSP dans le préfixe OTMA. La définition de cet indicateur demande à OTMA/IMS de générer un message DFS2082 RESPONSE MODE TRANSACTION TERMINATED WITHOUT REPLY lorsque le programme d'application IMS d'origine ne répond pas au IOPCB ni ne bascule vers une autre transaction.

La valeur initiale de cette zone est MQIIH\_NONE.

### ***LTermOverride (MQCHAR8) pour MQIIH***

Remplacement du terminal logique, placé dans la zone IO PCB. Il est facultatif ; s'il n'est pas spécifié, le nom TPIPE est utilisé. Elle est ignorée si le premier octet est vide ou null.

La longueur de cette zone est indiquée par MQ\_LTERM\_OVERRIDE\_LENGTH. La valeur initiale de cette zone est de 8 caractères blancs.

### ***MFSMapName (MQCHAR8) pour MQIIH***

Nom de mappe des services de format de message, placé dans la zone IO PCB. Il est facultatif. En entrée, il représente le MID, en sortie, il représente le MOD. Il est ignoré si le premier octet est vide ou nul.

La longueur de cette zone est indiquée par MQ\_MFS\_MAP\_NAME\_LENGTH. La valeur initiale de cette zone est de 8 caractères blancs.

### ***Format ReplyTo(MQCHAR8) pour MQIIH***

Il s'agit du nom de format MQ du message de réponse envoyé en réponse au message en cours. La longueur de cette zone est indiquée par MQ\_FORMAT\_LENGTH. La valeur initiale de cette zone est MQFMT\_NONE.

Pour convertir les données du message de réponse à l'aide de MQGMO\_CONVERT, spécifiez MQIIH.replyToFormat= MQFMT\_STRING ou MQIIH.replyToFormat= MQFMT\_IMS\_VAR\_STRING. Pour plus d'informations sur l'utilisation de ces zones, voir «Format (MQCHAR8) pour MQMD», à la page 465.

Si la valeur par défaut (MQIIH.replyToFormat= MQFMT\_NONE) est utilisée sur le message de demande et que le message de réponse est extrait à l'aide de MQGMO\_CONVERT, aucune conversion de données n'est effectuée.

### ***Authentificateur (MQCHAR8) pour MQIIH***

Il s'agit du mot de passe RACF ou PassTicket. Elle est facultative ; si elle est spécifiée, elle est utilisée avec l'ID utilisateur dans le contexte de sécurité MQMD pour générer un UTOKEN envoyé à IMS afin de fournir un contexte de sécurité. S'il n'est pas spécifié, l'ID utilisateur est utilisé sans vérification. Cela dépend des paramètres des commutateurs RACF , qui peuvent nécessiter la présence d'un authentificateur.

Cette valeur est ignorée si le premier octet est vide ou null. La valeur spéciale suivante peut être utilisée:

#### **MQIAUT\_AUCUN**

Pas d'authentification.

Pour le langage de programmation C, la constante MQIAUT\_NONE\_ARRAY est également définie ; elle a la même valeur que MQIAUT\_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La longueur de cette zone est indiquée par MQ\_AUTHENTICATOR\_LENGTH. La valeur initiale de cette zone est MQIAUT\_NONE.

### ***ID TranInstance(MQBYTE16) pour MQIIH***

Il s'agit de l'identificateur de l'instance de transaction. Cette zone est utilisée par les messages de sortie provenant de IMS. Par conséquent, elle est ignorée lors de la première entrée. Si vous affectez à *TranState* la valeur MQITS\_IN\_CONVERSATION, celle-ci doit être fournie dans l'entrée suivante, ainsi que dans toutes les entrées suivantes, pour permettre à IMS de corrélérer les messages à la conversation correcte. Vous pouvez utiliser la valeur spéciale suivante:

#### **MQITII\_AUCUN**

Aucun identificateur d'instance de transaction.

Pour le langage de programmation C, la constante MQITII\_NONE\_ARRAY est également définie ; elle a la même valeur que MQITII\_NONE, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

La longueur de cette zone est indiquée par MQ\_TRAN\_INSTANCE\_ID\_LENGTH. La valeur initiale de cette zone est MQITII\_NONE.

### ***TranState (MQCHAR) pour MQIIH***

Indique l'état de la conversation IMS . Elle est ignorée lors de la première entrée car aucune conversation n'existe. Sur les entrées suivantes, il indique si une conversation est active ou non. En sortie, il est défini par IMS. Il doit s'agir de l'une des valeurs suivantes :

#### **MQITS\_DANS\_CONVERSATION**

En conversation.


#### **MQITS\_NON\_CONVERSATION**

Pas en conversation.

#### **MQITS\_ARCHITECTURE**

Renvoie les données d'état de transaction sous forme structurée.

Cette valeur est utilisée uniquement avec la commande IMS /DISPLAY TRAN . Elle renvoie les données d'état de la transaction dans la forme IMS structurée au lieu de la forme de caractères.

 Pour plus d'informations, voir [Ecriture de programmes de transaction IMS via IBM MQ](#).

La valeur initiale de cette zone est MQITS\_NOT\_IN\_CONVERSATION.

### **CommitMode (MQCHAR) pour MQIIH**

Il s'agit du mode de validation IMS . Pour plus d'informations sur les modes de validation IMS , voir *OTMA Reference* . Il doit s'agir de l'une des valeurs suivantes :

#### **MQICM\_COMMIT\_THEN\_SEND**

Validez puis envoyez.

Ce mode implique une double mise en file d'attente de la sortie, mais des temps d'occupation de région plus courts. Les transactions rapides et conversationnelles ne peuvent pas s'exécuter avec ce mode.

#### **MQICM\_SEND\_THEN\_COMMIT**

Envoyer puis valider.

Toute transaction IMS lancée suite à un mode de validation de MQICM\_SEND\_THEN\_COMMIT s'exécute en mode RESPONSE, quelle que soit la manière dont la transaction est définie dans la définition du système IMS (paramètre MSGTYPE dans la macro TRANSACT). Cela s'applique également aux transactions initiées au moyen d'un commutateur de transaction.

La valeur initiale de cette zone est MQICM\_COMMIT\_THEN\_SEND.

### **SecurityScope (MQCHAR) pour MQIIH**

Indique le traitement de sécurité IMS requis. Les valeurs suivantes sont définies :

#### **CONTROLE MQISS\_CHECK**

Vérifier la portée de la sécurité: un ACEE est généré dans la région de contrôle, mais pas dans la région dépendante.

#### **MQISS\_FULL**

Portée de sécurité complète: un ACEE mis en cache est généré dans la région de contrôle et un ACEE non mis en cache est généré dans la région dépendante. Si vous utilisez MQISS\_FULL, vérifiez que l'ID utilisateur pour lequel ACEE est généré a accès aux ressources utilisées dans la région dépendante.

Si ni MQISS\_CHECK ni MQISS\_FULL n'est spécifié pour cette zone, MQISS\_CHECK est utilisé par défaut.

La valeur initiale de cette zone est MQISS\_CHECK.

### **Réservé (MQCHAR) pour MQIIH**

Il s'agit d'une zone réservée ; elle doit être vide.

### **MQIMPO-Options de propriété de message d'interrogation**

La structure MQIMPO permet aux applications de spécifier des options qui contrôlent la manière dont les propriétés des messages sont recherchées. La structure est un paramètre d'entrée dans l'appel MQINQMP.

### **Disponibilité**

Tous les systèmes IBM MQ et clients IBM MQ .

### **Jeu de caractères et codage**

Les données de MQIMPO doivent être dans le jeu de caractères de l'application et dans le codage de l'application (MQENC\_NATIVE).

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Tableau 498. Zones dans MQIPMO		
Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
StrucId (identificateur de structure)	ID_STRUC_MQIMPO_	'IMPO'
Version (numéro de version de la structure)	MQIMPO_VERSION_1	1
Options (options contrôlant l'action de MQINQMP)	MQIMPO_INQ_FIRST	
RequestedEncoding (codage dans lequel la propriété inquired doit être convertie)	MQENC_NATIVE	
RequestedCCSID (jeu de caractères de la propriété inquired)	MQCCSI_APPL	
ReturnedEncoding (codage de la valeur renvoyée)	MQENC_NATIVE	
ReturnedCCSID	0	
Reserved1 (zone réservée)	caractère blanc (zone de 4 octets)	
ReturnedName (nom de la propriété inquired)	MQCHARV_VALEUR PAR DEFAULT	
TypeString (représentation sous forme de chaîne du type de données de la propriété)	Chaîne nulle ou blancs	
<p><b>Remarques :</b></p> <ol style="list-style-type: none"> <li>1. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.</li> <li>2. Dans le langage de programmation C, la variable macroMQIMPO_DEFAULT contient les valeurs répertoriées dans la table. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</li> </ol> <pre style="background-color: #f0f0f0; padding: 10px;">MQIMPO MyIMPO = {MQIMPO_DEFAULT};</pre>		

## Déclarations de langue

Déclaration C pour MQIMPO

```
typedef struct tagMQIMPO MQIMPO;
struct tagMQIMPO {
    MQCHAR4  StructId;           /* Structure identifier */
    MQLONG   Version;           /* Structure version number */
    MQLONG   Options;           /* Options that control the action of
                                MQINQMP */
    MQLONG   RequestedEncoding; /* Requested encoding of Value */
    MQLONG   RequestedCCSID;    /* Requested character set identifier
                                of Value */
    MQLONG   ReturnedEncoding;  /* Returned encoding of Value */
    MQLONG   ReturnedCCSID;     /* Returned character set identifier
                                of Value */
    MQCHAR   Reserved1;        /* Reserved field */
    MQCHARV  ReturnedName;     /* Returned property name */
};
```

```

MQCHAR8 TypeString;          /* Property data type as a string */
};

```

## Déclaration COBOL pour MQIMPO

```

** MQIMPO structure
10 MQIMPO.
**   Structure identifier
15 MQIMPO-STRUCID          PIC X(4).
**   Structure version number
15 MQIMPO-VERSION        PIC S9(9) BINARY.
**   Options that control the action of MQINQMP
15 MQIMPO-OPTIONS        PIC S9(9) BINARY.
**   Requested encoding of VALUE
15 MQIMPO-REQUESTEDENCODING PIC S9(9) BINARY.
**   Requested character set identifier of VALUE
15 MQIMPO-REQUESTEDCCSID  PIC S9(9) BINARY.
**   Returned encoding of VALUE
15 MQIMPO-RETURNEDENCODING PIC S9(9) BINARY.
**   Returned character set identifier of VALUE
15 MQIMPO-RETURNEDCCSID  PIC S9(9) BINARY.
**   Reserved field
15 MQIMPO-RESERVED1
**   Returned property name
15 MQIMPO-RETURNEDNAME.
**   Address of variable length string
20 MQIMPO-RETURNEDNAME-VSPTR  POINTER.
**   Offset of variable length string
20 MQIMPO-RETURNEDNAME-VSOFFSET PIC S9(9) BINARY.
**   CCSID of variable length string
20 MQIMPO-RETURNEDNAME-VSCCSID PIC S9(9) BINARY.
**   Property data type as string
15 MQIMPO-TYPESTRING        PIC S9(9) BINARY.

```

## Déclaration PL/I pour MQIMPO

```

dcl
  1 MQIMPO based,
  3 StrucId          char(4),          /* Structure identifier */
  3 Version          fixed bin(31),    /* Structure version number */
  3 Options          fixed bin(31),    /* Options that control the
                                     action of MQINQMP */
  3 RequestedEncoding fixed bin(31),  /* Requested encoding of
                                     Value */
  3 RequestedCCSID   fixed bin(31),  /* Requested character set
                                     identifier of Value */
  3 ReturnedEncoding fixed bin(31),  /* Returned encoding of
                                     Value */
  3 ReturnedCCSID    fixed bin(31),  /* Returned character set
                                     identifier of Value */
  3 Reserved1        fixed bin(31),  /* Reserved field */
  3 ReturnedName,    /* Returned property name */
  5 ReturnedName_VSPtr  pointer,      /* Address of returned
                                     name */
  5 5 ReturnedName_VSOffset fixed bin(31), /* Offset of returned
                                     name */
  5 5 ReturnedName_VSCCSID fixed bin(31), /* CCSID of returned
                                     name */
  3 TypeString        char(8);        /* Property data type as
                                     string */

```

## Déclaration High Level Assembler pour MQIMPO

MQIMPO	DSECT	
MQIMPO_STRUCID	DS	CL4 Structure identifier
MQIMPO_VERSION	DS	F Structure version number
MQIMPO_OPTIONS	DS	F Options that control the
*		action of MQINQMP
MQIMPO_REQUESTEDENCODING	DS	F Requested encoding of VALUE
MQIMPO_REQUESTEDCCSID	DS	F Requested character set
*		identifier of VALUE
MQIMPO_RETURNEDENCODING	DS	F Returned encoding of VALUE
MQIMPO_RETURNEDCCSID	DS	F Returned character set
*		identifier of VALUE
MQIMPO_RESERVED1	DS	F Reserved field

MQIMPO_RETURNEDNAME	DS	0F	Force fullword alignment
MQIMPO_RETURNEDNAME_VSPTR	DS	F	Address of returned name
MQIMPO_RETURNEDNAME_VSOFFSET	DS	F	Offset of returned name
MQIMPO_RETURNEDNAME_VSLENGTH	DS	F	Length of returned name
MQIMPO_RETURNEDNAME_VSCCSID	DS	F	CCSID of returned name
MQIMPO_RETURNEDNAME_LENGTH	EQU	*-MQIMPO_RETURNEDNAME	
	ORG	MQIMPO_RETURNEDNAME	
MQIMPO_RETURNEDNAME_AREA	DS	CL(MQIMPO_RETURNEDNAME_LENGTH)	
*			
MQIMPO_TYPESTRING	DS	CL8	Property data type as string
MQIMPO_LENGTH	EQU	*-MQIMPO	
MQIMPO_AREA	DS	CL(MQIMPO_LENGTH)	

### **StrucId (MQCHAR4) pour MQIMPO**

Il s'agit de l'identificateur de structure de la structure d'options de propriété de message d'interrogation. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQIMPO\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUC\_MQIMPO\_**

Identificateur de la structure des options de propriété de message d'interrogation.

Pour le langage de programmation C, la constante MQIMPO\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQIMPO\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQIMPO**

Consulter la structure des options de propriété de message-Zone Version

Il s'agit du numéro de version de la structure. La valeur doit être:

#### **MQIMPO\_VERSION\_1**

Numéro de version de la structure des options de propriété de message d'interrogation.

La constante suivante indique le numéro de version de la version en cours:

#### **MQIMPO\_CURRENT\_VERSION**

Version actuelle de la structure des options de propriété de message d'interrogation.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQIMPO\_VERSION\_1.

### **Options (MQLONG) pour MQIMPO**

Structure des options d'interrogation des propriétés de message-Zone Options

Les options suivantes contrôlent l'action de MQINQMP. Vous pouvez spécifier une ou plusieurs de ces options. Pour spécifier plusieurs options, ajoutez les valeurs ensemble (n'ajoutez pas la même constante plusieurs fois) ou combinez les valeurs à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations bit).

Les combinaisons d'options non valides sont notées ; toutes les autres combinaisons sont valides.

**Options de données de valeur:** Les options suivantes concernent le traitement des données de valeur lorsque la propriété est extraite du message.

#### **MQIMPO\_CONVERT\_VALUE**

Cette option demande que la valeur de la propriété soit convertie pour être conforme aux valeurs *RequestedCCSID* et *RequestedEncoding* spécifiées avant que l'appel MQINQMP ne renvoie la valeur de la propriété dans la zone *Value* .

- Si la conversion aboutit, les zones *ReturnedCCSID* et *ReturnedEncoding* sont définies sur les mêmes valeurs que *RequestedCCSID* et *RequestedEncoding* en cas de retour de l'appel MQINQMP.
- Si la conversion échoue, mais que l'appel MQINQMP se termine sans erreur, la valeur de la propriété est renvoyée non convertie.



Si la propriété est une chaîne, les zones *ReturnedCCSID* et *ReturnedEncoding* sont définies sur le jeu de caractères et le codage de la chaîne non convertie.

Le code achèvement est MQCC\_WARNING dans ce cas, avec le code anomalie MQRC\_PROP\_VALUE\_NOT\_CONVERTED. Le curseur de propriété est avancé vers la propriété renvoyée.

Si la valeur de la propriété se développe lors de la conversion et dépasse la taille du paramètre **Value**, la valeur est renvoyée non convertie, avec le code achèvement MQCC\_FAILED ; le code anomalie est défini sur MQRC\_PROPERTY\_VALEUR\_TOO\_BIG.

Le paramètre **DataLength** de l'appel MQINQMP renvoie la longueur à laquelle la valeur de propriété aurait été convertie, afin de permettre à l'application de déterminer la taille de la mémoire tampon requise pour prendre en charge la valeur de propriété convertie. Le curseur de propriété est inchangé.

Cette option demande également que:

- Si le nom de la propriété contient un caractère générique, et
- La zone *ReturnedName* est initialisée avec une adresse ou un décalage pour le nom renvoyé,

le nom renvoyé est converti pour être conforme aux valeurs *RequestedCCSID* et *RequestedEncoding*.

- Si la conversion aboutit, la zone *VSCCSID* de *ReturnedName* et le codage du nom renvoyé sont définis sur la valeur d'entrée *RequestedCCSID* et *RequestedEncoding*.
- Si la conversion échoue, mais que l'appel MQINQMP se termine sans erreur ni avertissement, le nom renvoyé n'est pas converti. Le code achèvement est MQCC\_WARNING dans ce cas, avec le code anomalie MQRC\_PROP\_NAME\_NOT\_CONVERTIS.

Le curseur de propriété est avancé vers la propriété renvoyée.

MQRC\_PROP\_VALEUR\_NOT\_CONVERTED est renvoyé si la valeur et le nom ne sont pas convertis.

Si le nom renvoyé se développe lors de la conversion et dépasse la taille de la zone *VSBuFSIZE* de la *RequestedName*, la chaîne renvoyée reste non convertie, avec le code achèvement MQCC\_FAILED et le code anomalie défini sur MQRC\_PROPERTY\_NAME\_TOO\_BIG.

La zone *VSLength* de la structure MQCHARV renvoie la longueur à laquelle la valeur de propriété aurait été convertie, afin de permettre à l'application de déterminer la taille de la mémoire tampon requise pour accueillir la valeur de propriété convertie. Le curseur de propriété est inchangé.

### MQIMPO\_TYPE\_CONVERT

Cette option demande que la valeur de la propriété soit convertie de son type de données en cours en type de données spécifié dans le paramètre **Type** de l'appel MQINQMP.

- Si la conversion aboutit, le paramètre **Type** est inchangé lors du renvoi de l'appel MQINQMP.
- Si la conversion échoue, mais que l'appel MQINQMP se termine sans erreur, l'appel échoue avec la raison MQRC\_PROP\_CONV\_NOT\_SUPPORTED. Le curseur de propriété est inchangé.

Si la conversion du type de données entraîne l'extension de la valeur lors de la conversion et que la valeur convertie dépasse la taille du paramètre **Value**, la valeur est renvoyée non convertie avec le code achèvement MQCC\_FAILED et le code anomalie est défini sur MQRC\_PROPERTY\_VALEUR\_TOO\_BIG.

Le paramètre **DataLength** de l'appel MQINQMP renvoie la longueur à laquelle la valeur de propriété aurait été convertie, afin de permettre à l'application de déterminer la taille de la mémoire tampon requise pour prendre en charge la valeur de propriété convertie. Le curseur de propriété est inchangé.

Si la valeur du paramètre **Type** de l'appel MQINQMP n'est pas valide, l'appel échoue avec la raison MQRC\_PROPERTY\_TYPE\_ERROR.

Si la conversion de type de données demandée n'est pas prise en charge, l'appel échoue avec la raison MQRC\_PROP\_CONV\_NOT\_SUPPORTED. Les conversions de type de données suivantes sont prises en charge:

Tableau 499. Conversions de type de données prises en charge

Type de données de propriété	Types de données cible pris en charge
MQTYPE_BOOLÉEN	MQTYPE_STRING, MQTYPE_INT8, MQTYPE_INT16, MQTYPE_INT32, MQTYPE_INT64
MQTYPE_BYTE_STRING	MQTYPE_CHAINE
MQTYPE_INT8	MQTYPE_STRING, MQTYPE_INT16, MQTYPE_INT32, MQTYPE_INT64
MQTYPE_INT16	MQTYPE_STRING, MQTYPE_INT32, MQTYPE_INT64
MQTYPE_INT32	MQTYPE_STRING, MQTYPE_INT64
MQTYPE_INT64	MQTYPE_CHAINE
MQTYPE_FLOAT32	MQTYPE_STRING, MQTYPE_FLOAT64
MQTYPE_FLOAT64	MQTYPE_CHAINE
MQTYPE_CHAINE	MQTYPE_BOOLEAN, MQTYPE_INT8, MQTYPE_INT16, MQTYPE_INT32, MQTYPE_INT64, MQTYPE_FLOAT32, MQTYPE_FLOAT64
MQTYPE_NULL	Aucun

Les règles générales régissant les conversions prises en charge sont les suivantes:

- Les valeurs de propriété numérique peuvent être converties d'un type de données à un autre, à condition qu'aucune donnée ne soit perdue lors de la conversion.

Par exemple, la valeur d'une propriété avec le type de données MQTYPE\_INT32 peut être convertie en valeur avec le type de données MQTYPE\_INT64, mais ne peut pas être convertie en valeur avec le type de données MQTYPE\_INT16.

- Une valeur de propriété de tout type de données peut être convertie en chaîne.
- Une valeur de propriété de type chaîne peut être convertie en tout autre type de données, du moment que la chaîne est correctement formatée pour la conversion. Si une application tente de convertir une valeur de propriété de chaîne qui n'est pas formatée correctement, IBM MQ renvoie le code anomalie MQRC\_PROP\_NUMBER\_FORMAT\_ERROR.
- Si une application tente une conversion qui n'est pas prise en charge, IBM MQ renvoie le code anomalie MQRC\_PROP\_CONV\_NOT\_SUPPORTED.

Les règles spécifiques pour la conversion d'une valeur de propriété d'un type de données à un autre sont les suivantes:

- Lors de la conversion d'une valeur de propriété MQTYPE\_BOOLEAN en chaîne, la valeur TRUE est convertie en chaîne "TRUE" et la valeur false est convertie en chaîne "FALSE".
- Lors de la conversion d'une valeur de propriété MQTYPE\_BOOLEAN en un type de données numérique, la valeur TRUE est convertie en un et la valeur FALSE est convertie en zéro.
- Lors de la conversion d'une valeur de propriété de chaîne en valeur MQTYPE\_BOOLEAN, la chaîne "TRUE", ou "1", est convertie en TRUE, et la chaîne "FALSE", ou "0", est convertie en FALSE.

Notez que les termes "TRUE" et "FALSE" ne sont pas sensibles à la casse.

Toute autre chaîne ne peut pas être convertie ; IBM MQ renvoie le code anomalie MQRC\_PROP\_NUMBER\_FORMAT\_ERROR.

- Lors de la conversion d'une valeur de propriété de chaîne en valeur avec le type de données MQTYPE\_INT8, MQTYPE\_INT16, MQTYPE\_INT32 ou MQTYPE\_INT64, la chaîne doit avoir le format suivant:

```
[blanks][sign]digits
```

Les significations des composants de la chaîne sont les suivantes:

**blanks**

Caractères blancs de début facultatifs

**sign**

Caractère facultatif de signe plus (+) ou moins (-).

**digits**

Séquence contiguë de caractères numériques (0-9). Au moins un chiffre doit être présent.

Après la séquence de caractères numériques, la chaîne peut contenir d'autres caractères autres que des chiffres, mais la conversion s'arrête au premier caractère de ce type. La chaîne est supposée représenter un entier décimal.

IBM MQ renvoie le code anomalie MQRC\_PROP\_NUMBER\_FORMAT\_ERROR si la chaîne n'est pas correctement formatée.

- Lors de la conversion d'une valeur de propriété de chaîne en valeur avec le type de données MQTYPE\_FLOAT32 ou MQTYPE\_FLOAT64, la chaîne doit avoir le format suivant:

```
[blanks][sign]digits[.digits][e_char[e_sign]e_digits]
```

Les significations des composants de la chaîne sont les suivantes:

**blanks**

Caractères blancs de début facultatifs

**sign**

Caractère facultatif de signe plus (+) ou moins (-).

**digits**

Séquence contiguë de caractères numériques (0-9). Au moins un chiffre doit être présent.

**e\_char**

Un caractère exposant, qui est soit "E", soit "e".

**e\_sign**

Signe plus (+) ou signe moins (-) facultatif pour l'exposant.

**e\_digits**

Séquence contiguë de caractères numériques (0-9) pour l'exposant. Au moins un chiffre doit être présent si la chaîne contient un caractère exposant.

Après la séquence de chiffres, la chaîne peut contenir d'autres caractères autres que des chiffres, mais la conversion s'arrête au premier caractère de ce type. La chaîne est supposée représenter un nombre décimal en virgule flottante avec un exposant puissance de 10.

IBM MQ renvoie le code anomalie MQRC\_PROP\_NUMBER\_FORMAT\_ERROR si la chaîne n'est pas correctement formatée.

- Lors de la conversion d'une valeur de propriété numérique en chaîne, la valeur est convertie en représentation de chaîne de la valeur sous forme de nombre décimal, et non en chaîne contenant le caractère ASCII pour cette valeur. Par exemple, l'entier 65 est converti en chaîne "65", et non en chaîne "A".
- Lors de la conversion d'une valeur de propriété de chaîne d'octets en chaîne, chaque octet est converti en deux caractères hexadécimaux représentant l'octet. Par exemple, le tableau d'octets {0xF1, 0x12, 0x00, 0xFF} est converti en chaîne "F11200FF".

### **LONGUEUR\_REQUÊTE\_MQIMPO\_LONG**

Interrogez le type et la longueur de la valeur de propriété. La longueur est renvoyée dans le paramètre **DataLength** de l'appel MQINQMP. La valeur de la propriété n'est pas renvoyée.

Si une mémoire tampon **ReturnedName** est spécifiée, la zone *VSLength* de la structure MQCHARV est renseignée avec la longueur du nom de propriété. Le nom de propriété n'est pas renvoyé.

**Options d'itération:** Les options suivantes concernent l'itération sur les propriétés, à l'aide d'un nom avec un caractère générique

### **MQIMPO\_INQ\_FIRST**

Renseignez-vous sur la première propriété qui correspond au nom spécifié. Après cet appel, un curseur est établi sur la propriété renvoyée.

Il s'agit de la valeur par défaut.

L'option MQIMPO\_INQ\_PROP\_UNDER\_CURSOR peut ensuite être utilisée avec un appel MQINQMP, si nécessaire, pour demander à nouveau la même propriété.

Notez qu'il n'y a qu'un seul curseur de propriété ; par conséquent, si le nom de propriété, spécifié dans l'appel MQINQMP, change le curseur est réinitialisé.

Cette option n'est pas valide avec l'une des options suivantes:

MQIMPO\_INQ\_NEXT  
MQIMPO\_INQ\_PROP\_UNDER\_CURSOR

### **MQIMPO\_INQ\_NEXT**

Demande sur la propriété suivante qui correspond au nom spécifié, en poursuivant la recherche à partir du curseur de propriété. Le curseur est avancé à la propriété qui est renvoyée.

S'il s'agit du premier appel MQINQMP pour le nom spécifié, la première propriété correspondant au nom spécifié est renvoyée.

L'option MQIMPO\_INQ\_PROP\_UNDER\_CURSOR peut ensuite être utilisée avec un appel MQINQMP, si nécessaire, pour demander à nouveau la même propriété.

Si la propriété sous le curseur a été supprimée, MQINQMP renvoie la propriété correspondante suivante après celle qui a été supprimée.

Si une propriété qui correspond au caractère générique est ajoutée alors qu'une itération est en cours, elle peut ou non être renvoyée lors de la fin de l'itération. La propriété est renvoyée une fois que l'itération redémarre à l'aide de MQIMPO\_INQ\_FIRST.

Une propriété correspondant au caractère générique qui a été supprimé alors que l'itération était en cours n'est pas renvoyée après sa suppression.

Cette option n'est pas valide avec l'une des options suivantes:

MQIMPO\_INQ\_FIRST  
MQIMPO\_INQ\_PROP\_UNDER\_CURSOR

### **MQIMPO\_INQ\_PROP\_UNDER\_CURSOR**

Extrayez la valeur de la propriété pointée par le curseur de propriété. La propriété pointée par le curseur de propriété est celle qui a été interrogée pour la dernière fois, à l'aide de l'option MQIMPO\_INQ\_FIRST ou MQIMPO\_INQ\_NEXT.

Le curseur de propriété est réinitialisé lorsque le descripteur de message est réutilisé, lorsque le descripteur de message est spécifié dans la zone *MsgHandle* du MQGMO sur un appel MQGET, ou lorsque le descripteur de message est spécifié dans les zones *OriginalMsgHandle* ou *NewMsgHandle* de la structure MQPMO sur un appel MQPUT.

Si cette option est utilisée lorsque le curseur de propriété n'a pas encore été établi ou si la propriété pointée par le curseur de propriété a été supprimée, l'appel échoue avec le code achèvement MQCC\_FAILED et la raison MQRC\_PROPERTY\_NOT\_AVAILABLE.

Cette option n'est pas valide avec l'une des options suivantes:

MQIMPO\_INQ\_FIRST  
MQIMPO\_INQ\_NEXT

Si aucune des options décrites précédemment n'est requise, l'option suivante peut être utilisée:

### **MQIMPO\_NONE**

Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. Toutes les options sont définies sur leur valeur par défaut.

MQIMPO\_NONE aide la documentation du programme ; il n'est pas prévu que cette option soit utilisée avec d'autres, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQIMPO\_INQ\_FIRST.

### ***RequestedEncoding (MQLONG) pour MQIMPO***

Interroger la structure des options de propriété de message-Zone RequestedEncoding

Il s'agit du codage dans lequel la valeur de propriété interrogée doit être convertie lorsque MQIMPO\_CONVERT\_VALUE ou MQIMPO\_CONVERT\_TYPE est spécifié.

La valeur initiale de cette zone est MQENC\_NATIVE.

### ***RequestedCCSID (MQLONG) pour MQIMPO***

Structure des options d'interrogation des propriétés de message-Zone RequestedCCSID

Jeu de caractères dans lequel la valeur de la propriété recherchée doit être convertie si la valeur est une chaîne de caractères. Il s'agit également du jeu de caractères dans lequel le *ReturnedName* doit être converti lorsque MQIMPO\_CONVERT\_VALUE ou MQIMPO\_CONVERT\_TYPE est spécifié.

La valeur initiale de cette zone est MQCCSI\_APPL.

### ***ReturnedEncoding (MQLONG) pour MQIMPO***

Consulter la structure des options de propriété de message- ReturnedEncoding

En sortie, il s'agit du codage de la valeur renvoyée.

Si l'option MQIMPO\_CONVERT\_VALUE est spécifiée et que la conversion a abouti, la zone *ReturnedEncoding* , en retour, est identique à la valeur transmise.

La valeur initiale de cette zone est MQENC\_NATIVE.

### ***ReturnedCCSID (MQLONG) pour MQIMPO***

Interroger la structure des options de propriété de message-Zone ReturnedCCSID

En sortie, il s'agit du jeu de caractères de la valeur renvoyée si le paramètre **Type** de l'appel MQINQMP est MQTYPE\_STRING.

Si l'option MQIMPO\_CONVERT\_VALUE est spécifiée et que la conversion a abouti, la zone *ReturnedCCSID* , en retour, est identique à la valeur transmise.

La valeur initiale de cette zone est zéro.

### ***Reserved1 (MQCHAR) pour MQIMPO***

Il s'agit d'une zone réservée. La valeur initiale de cette zone est un caractère blanc (zone de 4 octets).

### ***ReturnedName (MQCHARV) pour MQIMPO***

Interroger la structure des options de propriété de message-Zone ReturnedName

Nom réel de la propriété interrogée.

En entrée, une mémoire tampon de chaîne peut être transmise à l'aide de la zone *VSPtr* ou *VSOffset* de la structure MQCHARV . La longueur de la mémoire tampon de chaîne est spécifiée à l'aide de la zone *VSBufsize* de la structure MQCHARV.

Lors du retour de l'appel MQINQMP, la mémoire tampon de la chaîne est terminée avec le nom de la propriété qui a été interrogée, à condition que la mémoire tampon de la chaîne soit suffisamment longue pour contenir complètement le nom. La zone *VSLength* de la structure MQCHARV est renseignée avec la longueur du nom de propriété. La zone *VSCCSID* de la structure MQCHARV est renseignée pour indiquer le jeu de caractères du nom renvoyé, que la conversion du nom ait échoué ou non.

Il s'agit d'une zone d'entrée/sortie. La valeur initiale de cette zone est MQCHARV\_DEFAULT.

### **TypeString (MQCHAR8) pour MQIMPO**

Structure des options d'interrogation des propriétés de message-Zone TypeString

Représentation sous forme de chaîne du type de données de la propriété.

Si la propriété a été spécifiée dans un en-tête MQRFH2 et que l'attribut MQRFH2 dt n'est pas reconnu, cette zone peut être utilisée pour déterminer le type de données de la propriété. *TypeString* est renvoyé dans le jeu de caractères codés 1208 (UTF-8) et correspond aux huit premiers octets de la valeur de l'attribut dt de la propriété dont la reconnaissance a échoué.

Il s'agit toujours d'une zone de sortie. La valeur initiale de cette zone est la chaîne nulle dans le langage de programmation C et 8 caractères blancs dans les autres langages de programmation.

## **MQMD-Descripteur de message**

La structure MQMD contient les informations de contrôle qui accompagnent les données d'application lorsqu'un message circule entre les applications d'envoi et de réception. La structure est un paramètre d'entrée/sortie sur les appels MQGET, MQPUT et MQPUT1 .

### **Disponibilité**

Tous les systèmes IBM MQ , plus IBM MQ MQI clients connectés à ces systèmes.

### **Version**

La version actuelle de MQMD est MQMD\_VERSION\_2. Les applications destinées à être portables entre plusieurs environnements doivent garantir que la version requise de MQMD est prise en charge dans tous les environnements concernés. Les zones qui existent uniquement dans les versions les plus récentes de la structure sont identifiées comme telles dans les descriptions qui suivent.

Les fichiers d'en-tête, COPY et INCLUDE fournis pour les langages de programmation pris en charge contiennent la version la plus récente de MQMD prise en charge par l'environnement, mais avec la valeur initiale de la zone *Version* définie sur MQMD\_VERSION\_1. Pour utiliser des zones qui ne sont pas présentes dans la structure version-1 , l'application doit définir la zone *Version* sur le numéro de version de la version requise.

Une déclaration pour la structure version-1 est disponible avec le nom MQMD1.

### **Jeu de caractères et codage**

Les données de MQMD doivent être dans le jeu de caractères et le codage du gestionnaire de files d'attente local ; ces données sont fournies par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et MQENC\_NATIVE. Toutefois, si l'application s'exécute en tant que IBM MQ MQI client, la structure doit être dans le jeu de caractères et le codage du client.

Si les gestionnaires de files d'attente d'envoi et de réception utilisent des jeux de caractères ou des codages différents, les données de MQMD sont converties automatiquement. Il n'est pas nécessaire que l'application convertisse le MQMD.

## Utilisation de différentes versions de MQMD

Un MQMD version-2 équivaut à utiliser un MQMD version-1 et à préfixer les données de message avec une structure MQMDE. Toutefois, si toutes les zones de la structure MQMDE ont leurs valeurs par défaut, MQMDE peut être omis. Une instance version-1 de MQMD plus MQMDE est utilisée comme décrit ci-après:

- Dans les appels MQPUT et MQPUT1, si l'application fournit un MQMD version-1, l'application peut éventuellement préfixer les données de message avec un MQMDE, en définissant la zone *Format* de MQMD sur MQFMT\_MD\_EXTENSION pour indiquer qu'un MQMDE est présent. Si l'application ne fournit pas de MQMDE, le gestionnaire de files d'attente utilise les valeurs par défaut pour les zones du MQMDE.

**Remarque :** Plusieurs des zones qui existent dans le MQMD version-2 mais pas dans le MQMD version-1 sont des zones d'entrée/sortie sur les appels MQPUT et MQPUT1. Toutefois, le gestionnaire de files d'attente ne renvoie aucune valeur dans les zones équivalentes de MQMDE dans la sortie des appels MQPUT et MQPUT1 ; si l'application requiert ces valeurs de sortie, elle doit utiliser un MQMD version-2.

- Dans l'appel MQGET, si l'application fournit un MQMD version-1, le gestionnaire de files d'attente préfixe le message renvoyé avec un MQMDE, mais uniquement si une ou plusieurs des zones du MQMDE ont une valeur autre que la valeur par défaut. La zone *Format* de MQMD aura la valeur MQFMT\_MD\_EXTENSION pour indiquer qu'un MQMDE est présent.

Les valeurs par défaut utilisées par le gestionnaire de files d'attente pour les zones de MQMDE sont identiques aux valeurs initiales de ces zones, indiquées dans le [Tableau 503](#), à la page 494.

Lorsqu'un message se trouve dans une file d'attente de transmission, certaines zones de MQMD sont définies sur des valeurs particulières ; pour plus de détails, voir «[MQXQH-en-tête de file d'attente de transmission](#)», à la page 647.

## Contexte de message

Certaines zones de MQMD contiennent le contexte de message. Il existe deux types de contexte de message: *contexte d'identité* et *contexte d'origine*. Généralement :

- Le contexte d'identité est lié à l'application qui a *initialement* inséré le message
- Le contexte d'origine est lié à l'application qui a *le plus récemment* inséré le message.

Ces deux applications peuvent être la même application, mais elles peuvent également être des applications différentes (par exemple, lorsqu'un message est transmis d'une application à une autre).

Bien que le contexte d'identité et d'origine aient généralement la signification décrite, le contenu des deux types de zones de contexte dans MQMD dépend des options MQPMO\_\*\_CONTEXT qui sont spécifiées lors de l'insertion du message. Par conséquent, le contexte d'identité ne se rapporte pas nécessairement à l'application qui a initialement inséré le message, et le contexte d'origine ne se rapporte pas nécessairement à l'application qui a inséré le message le plus récemment ; il dépend de la conception de la suite d'applications.

L'agent MCA ne modifie jamais le contexte de message. Les agents MCA qui reçoivent des messages des gestionnaires de files d'attente éloignées utilisent l'option de contexte MQPMO\_SET\_ALL\_CONTEXT dans l'appel MQPUT ou MQPUT1. Cela permet à l'agent MCA récepteur de conserver exactement le contexte de message qui a été transmis avec le message à partir de l'agent MCA émetteur. Toutefois, il en résulte que le contexte d'origine ne se rapporte à aucun des agents MCA qui ont envoyé et reçu le message. Le contexte d'origine fait référence à une application antérieure qui a inséré le message. Si toutes les applications intermédiaires ont passé le contexte de message, le contexte d'origine fait référence à l'application d'origine elle-même.

Dans les descriptions, les zones de contexte sont décrites comme si elles étaient utilisées comme décrit précédemment. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#).

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

*Tableau 500. Zones de MQMD pour MQMD*

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID_STRUCD_MQM	'MD'
<u>Version</u> (numéro de version de la structure)	MQMD_VERSION_1	1
<u>Rapport</u> (options pour les messages de rapport)	MQRO_AUCUN	0
<u>MsgType</u> (type de message)	MQMT_DATAGRAM	8
<u>MQMD-Zone d'expiration</u> (durée de vie du message)	MQEI_UNLIMITED	-1
<u>MQMD-Zone de retour d'informations</u> (retour d'informations ou code anomalie)	MQFB_AUCUN	0
<u>Codage</u> (codage numérique des données de message)	MQENC_NATIVE	Dépend de l'environnement
<u>CodedCharSetId</u> (identificateur de jeu de caractères des données de message)	MQCCSI_Q_DIR	0
<u>Format</u> (nom de format des données de message)	MQFMT_AUCUN	Espaces vides
<u>Priorité</u> (priorité de message)	MQPRI_PRIORITY_AS_Q_DEF	-1
<u>Persistance</u> (persistance des messages)	MQPER_PERSISTENCE_AS_Q_DEF	2
<u>MQMD-Zone MsgId</u> (identificateur de message)	MQMI_AUCUN	Valeurs NULL
<u>CorrelId</u> (identificateur de corrélation)	MQCI_NONE	Valeurs NULL
<u>BackoutCount</u> (compteur d'annulation)	Aucun	0
<u>ReplyToQ</u> (nom de la file d'attente de réponses)	Aucun	Chaîne nulle ou blancs
<u>ReplyToGestionnaire de files d'attente</u> (nom du gestionnaire de files d'attente de réponses)	Aucun	Chaîne nulle ou blancs
<u>UserIdentifieur</u> (identificateur utilisateur)	Aucun	Chaîne nulle ou blancs
<u>AccountingToken</u> (jeton de comptabilité)	MQACT_AUCUN	Valeurs NULL
<u>DonnéesApplIdentity</u> (données d'application relatives à l'identité)	Aucun	Chaîne nulle ou blancs
<u>PutApplType</u> (type d'application qui a inséré le message)	MQAT_NO_CONTEXT	0
<u>PutApplNom</u> (nom de l'application qui a inséré le message)	Aucun	Chaîne nulle ou blancs
<u>PutDate</u> (date à laquelle le message a été inséré)	Aucun	Chaîne nulle ou blancs
<u>PutTime</u> (heure à laquelle le message a été inséré)	Aucun	Chaîne nulle ou blancs



Tableau 500. Zones de MQMD pour MQMD (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>DonnéesApplOrigin</u> (données d'application relatives à l'origine)	Aucun	Chaîne nulle ou blancs
<b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieur à MQMD_VERSION_2.		
<u>GroupId</u> (identificateur de groupe)	MQGI_AUCUN	Valeurs NULL
<u>MsgSeqNuméro</u> (numéro de séquence du message logique au sein du groupe)	Aucun	1
<u>Décalage</u> (décalage des données dans le message physique à partir du début du message logique)	Aucun	0
<u>MQMD-Zone MsgFlags</u> (indicateurs de message)	MQMF_AUCUN	0
<u>OriginalLength</u> (longueur du message d'origine)	MQOL_UNDEFINI	-1
<p><b>Remarques :</b></p> <ol style="list-style-type: none"> <li>1. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.</li> <li>2. Dans le langage de programmation C, la variable macroMQMD_DEFAULT contient les valeurs répertoriées dans la table. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</li> </ol> <pre style="background-color: #f0f0f0; padding: 10px;">MQMD MyMD = {MQMD_DEFAULT};</pre>		

## Déclarations de langue

Déclaration C pour MQMD

```
typedef struct tagMQMD MQMD;
struct tagMQMD {
    MQCHAR4   StructId;           /* Structure identifier */
    MQLONG    Version;           /* Structure version number */
    MQLONG    Report;           /* Options for report messages */
    MQLONG    MsgType;          /* Message type */
    MQLONG    Expiry;           /* Message lifetime */
    MQLONG    Feedback;         /* Feedback or reason code */
    MQLONG    Encoding;         /* Numeric encoding of message data */
    MQLONG    CodedCharSetId;    /* Character set identifier of message
    data */

    MQCHAR8   Format;           /* Format name of message data */
    MQLONG    Priority;         /* Message priority */
    MQLONG    Persistence;     /* Message persistence */
    MQBYTE24  MsgId;           /* Message identifier */
    MQBYTE24  CorrelId;        /* Correlation identifier */
    MQLONG    BackoutCount;    /* Backout counter */
    MQCHAR48  ReplyToQ;        /* Name of reply queue */
    MQCHAR48  ReplyToQMgr;     /* Name of reply queue manager */
    MQCHAR12  UserIdentifier;   /* User identifier */
    MQBYTE32  AccountingToken; /* Accounting token */
    MQCHAR32  ApplIdentityData; /* Application data relating to
    identity */

    MQLONG    PutApplType;     /* Type of application that put the
    message */
    MQCHAR28  PutApplName;    /* Name of application that put the
    message */

    MQCHAR8   PutDate;         /* Date when message was put */
    MQCHAR8   PutTime;         /* Time when message was put */
    MQCHAR4   ApplOriginData; /* Application data relating to origin */
}
```

```

MQBYTE24  GroupId;          /* Group identifier */
MQLONG    MsgSeqNumber;    /* Sequence number of logical message
                           within group */
MQLONG    Offset;          /* Offset of data in physical message
                           from start of logical message */
MQLONG    MsgFlags;        /* Message flags */
MQLONG    OriginalLength;  /* Length of original message */
};

```

## Déclaration COBOL pour MQMD

```

**  MQMD structure
  10 MQMD.
**  Structure identifier
  15 MQMD-STRUCID          PIC X(4).
**  Structure version number
  15 MQMD-VERSION        PIC S9(9) BINARY.
**  Options for report messages
  15 MQMD-REPORT         PIC S9(9) BINARY.
**  Message type
  15 MQMD-MSGTYPE        PIC S9(9) BINARY.
**  Message lifetime
  15 MQMD-EXPIRY         PIC S9(9) BINARY.
**  Feedback or reason code
  15 MQMD-FEEDBACK       PIC S9(9) BINARY.
**  Numeric encoding of message data
  15 MQMD-ENCODING       PIC S9(9) BINARY.
**  Character set identifier of message data
  15 MQMD-CODEDCHARSETID PIC S9(9) BINARY.
**  Format name of message data
  15 MQMD-FORMAT         PIC X(8).
**  Message priority
  15 MQMD-PRIORITY       PIC S9(9) BINARY.
**  Message persistence
  15 MQMD-PERSISTENCE    PIC S9(9) BINARY.
**  Message identifier
  15 MQMD-MSGID          PIC X(24).
**  Correlation identifier
  15 MQMD-CORRELID       PIC X(24).
**  Backout counter
  15 MQMD-BACKOUTCOUNT  PIC S9(9) BINARY.
**  Name of reply queue
  15 MQMD-REPLYTOQ       PIC X(48).
**  Name of reply queue manager
  15 MQMD-REPLYTOQMGR    PIC X(48).
**  User identifier
  15 MQMD-USERIDENTIFIER PIC X(12).
**  Accounting token
  15 MQMD-ACCOUNTINGTOKEN PIC X(32).
**  Application data relating to identity
  15 MQMD-APPLIDENTITYDATA PIC X(32).
**  Type of application that put the message
  15 MQMD-PUTAPPLTYPE    PIC S9(9) BINARY.
**  Name of application that put the message
  15 MQMD-PUTAPPLNAME    PIC X(28).
**  Date when message was put
  15 MQMD-PUTDATE        PIC X(8).
**  Time when message was put
  15 MQMD-PUTTIME        PIC X(8).
**  Application data relating to origin
  15 MQMD-APPLORIGINDATA PIC X(4).
**  Group identifier
  15 MQMD-GROUPID        PIC X(24).
**  Sequence number of logical message within group
  15 MQMD-MSGSEQNUMBER   PIC S9(9) BINARY.
**  Offset of data in physical message from start of logical message
  15 MQMD-OFFSET         PIC S9(9) BINARY.
**  Message flags
  15 MQMD-MSGFLAGS       PIC S9(9) BINARY.
**  Length of original message
  15 MQMD-ORIGINALLENGTH PIC S9(9) BINARY.

```

## Déclaration PL/I pour MQMD

```

dcl
  1 MQMD based,
  3 StrucId          char(4),          /* Structure identifier */

```

```

3 Version          fixed bin(31), /* Structure version number */
3 Report           fixed bin(31), /* Options for report messages */
3 MsgType          fixed bin(31), /* Message type */
3 Expiry           fixed bin(31), /* Message lifetime */
3 Feedback         fixed bin(31), /* Feedback or reason code */
3 Encoding         fixed bin(31), /* Numeric encoding of message
                                data */
3 CodedCharSetId  fixed bin(31), /* Character set identifier of
                                message data */
3 Format            char(8),      /* Format name of message data */
3 Priority          fixed bin(31), /* Message priority */
3 Persistence      fixed bin(31), /* Message persistence */
3 MsgId            char(24),     /* Message identifier */
3 CorrelId         char(24),     /* Correlation identifier */
3 BackoutCount     fixed bin(31), /* Backout counter */
3 ReplyToQ         char(48),     /* Name of reply queue */
3 ReplyToQMgr      char(48),     /* Name of reply queue manager */
3 UserIdentifier   char(12),     /* User identifier */
3 AccountingToken  char(32),     /* Accounting token */
3 ApplIdentityData char(32),     /* Application data relating to
                                identity */
3 PutApplType      fixed bin(31), /* Type of application that put the
                                message */
3 PutApplName      char(28),     /* Name of application that put the
                                message */
3 PutDate          char(8),      /* Date when message was put */
3 PutTime          char(8),      /* Time when message was put */
3 ApplOriginData  char(4),      /* Application data relating to
                                origin */
3 GroupId          char(24),     /* Group identifier */
3 MsgSeqNumber     fixed bin(31), /* Sequence number of logical
                                message within group */
3 Offset           fixed bin(31), /* Offset of data in physical
                                message from start of logical
                                message */
3 MsgFlags         fixed bin(31), /* Message flags */
3 OriginalLength   fixed bin(31); /* Length of original message */

```

## Déclaration High Level Assembler pour MQMD

```

MQMD              DSECT
MQMD_STRUCID      DS   CL4   Structure identifier
MQMD_VERSION      DS   F     Structure version number
MQMD_REPORT       DS   F     Options for report messages
MQMD_MSGTYPE      DS   F     Message type
MQMD_EXPIRY       DS   F     Message lifetime
MQMD_FEEDBACK     DS   F     Feedback or reason code
MQMD_ENCODING     DS   F     Numeric encoding of message data
MQMD_CODEDCHARSETID DS   F   Character set identifier of message
* data
MQMD_FORMAT       DS   CL8   Format name of message data
MQMD_PRIORITY     DS   F     Message priority
MQMD_PERSISTENCE  DS   F     Message persistence
MQMD_MSGID        DS   XL24  Message identifier
MQMD_CORRELID     DS   XL24  Correlation identifier
MQMD_BACKOUTCOUNT DS   F     Backout counter
MQMD_REPLYTOQ     DS   CL48  Name of reply queue
MQMD_REPLYTOQMGR  DS   CL48  Name of reply queue manager
MQMD_USERIDENTIFIER DS   CL12 User identifier
MQMD_ACCOUNTINGTOKEN DS   XL32 Accounting token
MQMD_APPLIDENTITYDATA DS   CL32 Application data relating to identity
MQMD_PUTAPPLTYPE  DS   F     Type of application that put the
* message
MQMD_PUTAPPLNAME  DS   CL28  Name of application that put the
* message
MQMD_PUTDATE      DS   CL8   Date when message was put
MQMD_PUTTIME      DS   CL8   Time when message was put
MQMD_APPLORIGINDATA DS   CL4  Application data relating to origin
MQMD_GROUPID      DS   XL24  Group identifier
MQMD_MSGSEQNUMBER DS   F     Sequence number of logical message
* within group
MQMD_OFFSET       DS   F     Offset of data in physical message
* from start of logical message
MQMD_MSGFLAGS     DS   F     Message flags
MQMD_ORIGINALLENGTH DS   F   Length of original message
*
MQMD_LENGTH       EQU   *-MQMD
                  ORG   MQMD
MQMD_AREA         DS   CL(MQMD_LENGTH)

```

## Déclaration Visual Basic pour MQMD

```
Type MQMD
  StructId      As String*4  'Structure identifier'
  Version       As Long      'Structure version number'
  Report        As Long      'Options for report messages'
  MsgType       As Long      'Message type'
  Expiry        As Long      'Message lifetime'
  Feedback      As Long      'Feedback or reason code'
  Encoding      As Long      'Numeric encoding of message data'
  CodedCharSetId As Long      'Character set identifier of message'
  'data'
  Format        As String*8  'Format name of message data'
  Priority      As Long      'Message priority'
  Persistence   As Long      'Message persistence'
  MsgId         As MQBYTE24  'Message identifier'
  CorrelId      As MQBYTE24  'Correlation identifier'
  BackoutCount  As Long      'Backout counter'
  ReplyToQ      As String*48  'Name of reply queue'
  ReplyToQMgr   As String*48  'Name of reply queue manager'
  UserIdentifier As String*12  'User identifier'
  AccountingToken As MQBYTE32 'Accounting token'
  ApplIdentityData As String*32 'Application data relating to identity'
  PutApplType   As Long      'Type of application that put the'
  'message'
  PutApplName   As String*28  'Name of application that put the'
  'message'
  PutDate       As String*8  'Date when message was put'
  PutTime       As String*8  'Time when message was put'
  ApplOriginData As String*4  'Application data relating to origin'
  GroupId       As MQBYTE24  'Group identifier'
  MsgSeqNumber  As Long      'Sequence number of logical message'
  'within group'
  Offset        As Long      'Offset of data in physical message'
  'from start of logical message'
  MsgFlags      As Long      'Message flags'
  OriginalLength As Long      'Length of original message'
End Type
```

### **StrucId (MQCHAR4) pour MQMD**

Il s'agit de l'identificateur de structure de la structure de descripteur de message. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQMD\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUCD\_MQM**

Identificateur de la structure de descripteur de message.

Pour le langage de programmation C, la constante MQMD\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQMD\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQMD**

Il s'agit du numéro de version de la structure, qui doit être l'un des suivants:

#### **MQMD\_VERSION\_1**

Structure de descripteur de message Version-1 .

Cette version est prise en charge dans tous les environnements.

#### **MQMD\_VERSION\_2**

Structure de descripteur de message Version-2 .

Cette version est prise en charge dans tous les environnements IBM MQ V6.0 et ultérieure, plus IBM MQ MQI clients connecté à ces systèmes.

**Remarque :** Lorsqu'un MQMD version-2 est utilisé, le gestionnaire de files d'attente effectue des vérifications supplémentaires sur les structures d'en-tête MQ qui peuvent être présentes au début des données de message d'application. Pour plus de détails, voir les remarques sur l'utilisation de l'appel MQPUT.

Les zones qui existent uniquement dans la version la plus récente de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

### **MQMD\_CURRENT\_VERSION**

Version actuelle de la structure de descripteur de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQMD\_VERSION\_1.

### **Rapport (MQLONG) pour MQMD**

Un message de rapport est un message relatif à un autre message, utilisé pour informer une application des événements attendus ou inattendus liés au message d'origine. La zone *Report* permet à l'application qui envoie le message d'origine de spécifier quels sont les messages de rapport requis, si les données de message d'application doivent y être incluses, ainsi que (pour les rapports et les réponses) la manière dont les identificateurs de message et de corrélation du message de rapport ou de réponse doivent être définis. Tout ou partie (ou aucun) des types de message de rapport suivants peut être demandé:

- Exception
- Expiration
- Confirmer à l'arrivée (COA)
- Confirmer à la livraison (COD)
- Notification d'action positive (PAN)
- Notification d'action négative (NAN)

Vous pouvez spécifier une ou plusieurs de ces options. Pour spécifier plusieurs options, ajoutez les valeurs ensemble (n'ajoutez pas la même constante plusieurs fois) ou combinez les valeurs à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations bit).

L'application qui reçoit le message de rapport peut déterminer la raison pour laquelle le rapport a été généré en examinant la zone *Feedback* dans le MQMD ; voir la zone *Feedback* pour plus de détails.

L'utilisation des options de rapport lors de l'insertion d'un message dans une rubrique peut entraîner la génération de zéro, un ou plusieurs messages de rapport et leur envoi à l'application. En effet, le message de publication peut être envoyé à zéro, une ou plusieurs applications d'abonnement.

**Options d'exception:** spécifiez l'une des options répertoriées pour demander un message de rapport d'exception.

### **MQRO\_EXCEPTION**

Un agent MCA génère ce type de rapport lorsqu'un message est envoyé à un autre gestionnaire de files d'attente et qu'il ne peut pas être distribué à la file d'attente de destination spécifiée. Par exemple, la file d'attente de destination ou une file d'attente de transmission intermédiaire peut être saturée ou le message peut être trop volumineux pour la file d'attente.

La génération du message de rapport d'exception dépend de la persistance du message d'origine et de la vitesse du canal de transmission des messages (normal ou rapide) par lequel le message d'origine est transmis:

- Pour tous les messages persistants et pour les messages non persistants qui transitent par des canaux de messages normaux, le rapport d'exception est généré uniquement si l'action spécifiée par l'application émettrice pour la condition d'erreur peut aboutir. L'application émettrice peut spécifier l'une des actions suivantes pour contrôler la disposition du message d'origine lorsque la condition d'erreur se produit:
  - MQRO\_DEAD\_LETTER\_Q (le message d'origine est placé dans la file d'attente des messages non livrés).
  - MQRO\_DISCARD\_MSG (supprime le message d'origine).

Si l'action spécifiée par l'application émettrice ne peut pas aboutir, le message d'origine est laissé dans la file d'attente de transmission et aucun message de rapport d'exception n'est généré.

- Pour les messages non persistants qui transitent par des canaux de messages rapides, le message d'origine est supprimé de la file d'attente de transmission et le rapport d'exception est généré *même si* l'action spécifiée pour la condition d'erreur ne peut pas aboutir. Par exemple, si MQRO\_DEAD\_LETTER\_Q est spécifié, mais que le message d'origine ne peut pas être placé dans la file d'attente de rebut car cette file d'attente est saturée, le message de rapport d'exception est généré et le message d'origine est supprimé.

Pour plus d'informations sur les canaux de messages normaux et rapides, voir [Vitesse de messages non persistants \(NPMSPEED\)](#).

Aucun rapport d'exception n'est généré si l'application qui a inséré le message d'origine peut être avertie de manière synchrone de l'incident au moyen du code anomalie renvoyé par l'appel MQPUT ou MQPUT1 .

Les applications peuvent également envoyer des rapports d'exception pour indiquer qu'un message ne peut pas être traité (par exemple, parce qu'il s'agit d'une transaction de débit qui entraînerait le dépassement de la limite de crédit du compte).

Les données du message d'origine ne sont pas incluses dans le message de rapport.

N'indiquez pas plus d'une seule des valeurs suivantes: MQRO\_EXCEPTION, MQRO\_EXCEPTION\_WITH\_DATA et MQRO\_EXCEPTION\_WITH\_FULL\_DATA.

#### **MQRO\_EXCEPTION\_WITH\_DATA**

Identique à MQRO\_EXCEPTION, sauf que les 100 premiers octets des données de message d'application du message d'origine sont inclus dans le message de rapport. Si le message d'origine contient une ou plusieurs structures d'en-tête MQ , elles sont incluses dans le message de rapport, en plus des 100 octets de données d'application.

N'indiquez pas plus d'une seule des valeurs suivantes: MQRO\_EXCEPTION, MQRO\_EXCEPTION\_WITH\_DATA et MQRO\_EXCEPTION\_WITH\_FULL\_DATA.

#### **MQRO\_EXCEPTION\_WITH\_FULL\_DATA**

Des rapports d'exception avec des données complètes sont requis.

Identique à MQRO\_EXCEPTION, sauf que toutes les données de message d'application du message d'origine sont incluses dans le message de rapport.

N'indiquez pas plus d'une seule des valeurs suivantes: MQRO\_EXCEPTION, MQRO\_EXCEPTION\_WITH\_DATA et MQRO\_EXCEPTION\_WITH\_FULL\_DATA.

**Options d'expiration:** spécifiez l'une des options répertoriées pour demander un message de rapport d'expiration.

#### **MQRO\_EXPIRATION**

Ce type de rapport est généré par le gestionnaire de files d'attente si le message est supprimé avant la livraison à une application en raison du délai d'expiration (voir la zone *Expiry* ). Si cette option n'est pas définie, aucun message de rapport n'est généré si un message est supprimé pour cette raison (même si vous spécifiez l'une des options MQRO\_EXCEPTION\_\*).

Les données du message d'origine ne sont pas incluses dans le message de rapport.

N'indiquez pas plus d'une des valeurs suivantes: MQRO\_EXPIRATION, MQRO\_EXPIRATION\_WITH\_DATA et MQRO\_EXPIRATION\_WITH\_FULL\_DATA.

#### **MQRO\_EXPIRATION\_WITH\_DATA**

Identique à MQRO\_EXPIRATION, sauf que les 100 premiers octets des données de message d'application du message d'origine sont inclus dans le message de rapport. Si le message d'origine contient une ou plusieurs structures d'en-tête MQ , elles sont incluses dans le message de rapport, en plus des 100 octets de données d'application.

N'indiquez pas plus d'une des valeurs suivantes: MQRO\_EXPIRATION, MQRO\_EXPIRATION\_WITH\_DATA et MQRO\_EXPIRATION\_WITH\_FULL\_DATA.

## **MQRO\_EXPIRATION\_WITH\_FULL\_DATA**

Identique à MQRO\_EXPIRATION, sauf que toutes les données de message d'application du message d'origine sont incluses dans le message de rapport.

N'indiquez pas plus d'une des valeurs suivantes: MQRO\_EXPIRATION, MQRO\_EXPIRATION\_WITH\_DATA et MQRO\_EXPIRATION\_WITH\_FULL\_DATA.

**Options de confirmation à l'arrivée:** indiquez l'une des options répertoriées pour demander un message de rapport de confirmation à l'arrivée.

## **MQRO\_COA**

Ce type de rapport est généré par le gestionnaire de files d'attente propriétaire de la file d'attente de destination lorsque le message est placé dans la file d'attente de destination. Les données du message d'origine ne sont pas incluses dans le message de rapport.

Si le message est inséré dans une unité de travail et que la file d'attente de destination est une file d'attente locale, le message de rapport COA généré par le gestionnaire de files d'attente peut être extrait uniquement si l'unité de travail est validée.

Aucun rapport COA n'est généré si la zone *Format* du descripteur de message est MQFMT\_XMIT\_Q\_HEADER ou MQFMT\_DEAD\_LETTER\_HEADER. Cela empêche la génération d'un rapport COA si le message est placé dans une file d'attente de transmission ou s'il n'est pas livrable et placé dans une file d'attente de rebut.

Dans le cas d'une file d'attente de pont IMS, le rapport COA est généré lorsque le message atteint la file d'attente IMS (accusé de réception reçu de IMS) et non lorsque le message est inséré dans la file d'attente de pont MQ. Cela signifie que si IMS n'est pas actif, aucun rapport COA n'est généré tant que IMS n'est pas démarré et qu'un message est mis en file d'attente dans la file d'attente IMS.

Utilisateur qui exécute un programme qui insère un message avec MQMD.Report= MQRO\_COA doit disposer des droits d'accès + passid sur la file d'attente de réponses. Si l'utilisateur ne dispose pas des droits + passid, le message de rapport COA n'atteint pas la file d'attente de réponses. Une tentative est effectuée pour placer le message de rapport dans la file d'attente des messages non livrés.

N'indiquez pas plus d'un des éléments MQRO\_COA, MQRO\_COA\_WITH\_DATA et MQRO\_COA\_WITH\_FULL\_DATA.

## **MQRO\_COA\_WITH\_DATA**

Il est identique à MQRO\_COA, sauf que les 100 premiers octets des données de message d'application du message d'origine sont inclus dans le message de rapport. Si le message d'origine contient une ou plusieurs structures d'en-tête MQ, elles sont incluses dans le message de rapport, en plus des 100 octets de données d'application.

N'indiquez pas plus d'un des éléments MQRO\_COA, MQRO\_COA\_WITH\_DATA et MQRO\_COA\_WITH\_FULL\_DATA.

## **MQRO\_COA\_WITH\_FULL\_DATA**

Identique à MQRO\_COA, sauf que toutes les données de message d'application du message d'origine sont incluses dans le message de rapport.

N'indiquez pas plus d'un des éléments MQRO\_COA, MQRO\_COA\_WITH\_DATA et MQRO\_COA\_WITH\_FULL\_DATA.

**Options de confirmation de livraison:** spécifiez l'une des options répertoriées pour demander un message de rapport de confirmation de livraison.

## **MQRO\_COD**

Ce type de rapport est généré par le gestionnaire de files d'attente lorsqu'une application extrait le message de la file d'attente de destination d'une manière qui supprime le message de la file d'attente. Les données du message d'origine ne sont pas incluses dans le message de rapport.

Si le message est extrait dans le cadre d'une unité de travail, le message de rapport est généré dans la même unité de travail, de sorte que le rapport n'est pas disponible tant que l'unité de travail n'est pas validée. Si l'unité de travail est annulée, le rapport n'est pas envoyé.

Un rapport COD n'est pas toujours généré si un message est extrait avec l'option MQGMO\_MARK\_SKIP\_BACKOUT. Si l'unité de travail principale est annulée mais que l'unité de travail secondaire est validée, le message est supprimé de la file d'attente, mais aucun rapport COD n'est généré.

Aucun rapport COD n'est généré si la zone *Format* du descripteur de message est MQFMT\_DEAD\_LETTER\_HEADER. Cela empêche la génération d'un rapport COD si le message n'est pas livrable et qu'il est placé dans une file d'attente de rebut.

MQRO\_COD n'est pas valide si la file d'attente de destination est une file d'attente XCF.

N'indiquez pas plus d'un MQRO\_COD, MQRO\_COD\_WITH\_DATA et MQRO\_COD\_WITH\_FULL\_DATA.

### **MQRO\_COD\_WITH\_DATA**

Cette valeur est identique à MQRO\_COD, sauf que les 100 premiers octets des données de message d'application du message d'origine sont inclus dans le message de rapport. Si le message d'origine contient une ou plusieurs structures d'en-tête MQ, elles sont incluses dans le message de rapport, en plus des 100 octets de données d'application.

Si MQGMO\_ACCEPT\_TRUNCATED\_MSG est spécifié dans l'appel MQGET pour le message d'origine et que le message extrait est tronqué, la quantité de données de message d'application placée dans le message de rapport dépend de l'environnement:

- Sous z/OS, il s'agit de la valeur minimale de:
  - Longueur du message d'origine
  - Longueur de la mémoire tampon utilisée pour extraire le message
  - 100 octets.
- Dans les autres environnements, il s'agit du minimum de:
  - Longueur du message d'origine
  - 100 octets.

MQRO\_COD\_WITH\_DATA n'est pas valide si la file d'attente de destination est une file d'attente XCF.

N'indiquez pas plus d'un MQRO\_COD, MQRO\_COD\_WITH\_DATA et MQRO\_COD\_WITH\_FULL\_DATA.

### **MQRO\_COD\_WITH\_FULL\_DATA**

Cette valeur est identique à MQRO\_COD, sauf que toutes les données de message d'application du message d'origine sont incluses dans le message de rapport.

MQRO\_COD\_WITH\_FULL\_DATA n'est pas valide si la file d'attente de destination est une file d'attente XCF.

N'indiquez pas plus d'un MQRO\_COD, MQRO\_COD\_WITH\_DATA et MQRO\_COD\_WITH\_FULL\_DATA.

**Options de notification d'action:** spécifiez l'une des options répertoriées ou les deux pour demander à l'application réceptrice d'envoyer un message de rapport d'action positive ou négative.

### **MQRO\_PAN**

Ce type de rapport est généré par l'application qui extrait le message et agit en conséquence. Elle indique que l'action demandée dans le message a été exécutée avec succès. L'application qui génère le rapport détermine si des données doivent être incluses dans le rapport.

A l'exception de la transmission de cette demande à l'application qui extrait le message, le gestionnaire de files d'attente n'effectue aucune action en fonction de cette option. L'application d'extraction doit générer le rapport le cas échéant.

### **MQRO\_NAN**

Ce type de rapport est généré par l'application qui extrait le message et agit en conséquence. Il indique que l'action demandée dans le message n'a pas été exécutée correctement. L'application qui génère le rapport détermine si des données doivent être incluses dans le rapport. Par exemple, vous pouvez inclure des données indiquant la raison pour laquelle la demande n'a pas pu être exécutée.



A l'exception de la transmission de cette demande à l'application qui extrait le message, le gestionnaire de files d'attente n'effectue aucune action en fonction de cette option. L'application d'extraction doit générer le rapport le cas échéant.

L'application doit déterminer les conditions qui correspondent à une action positive et celles qui correspondent à une action négative. Toutefois, si la requête n'a été que partiellement exécutée, générez un rapport NAN plutôt qu'un rapport PAN si demandé. Chaque condition possible doit correspondre soit à une action positive, soit à une action négative, mais pas les deux.

**Options d'identificateur de message:** spécifiez l'une des options répertoriées pour contrôler la manière dont le *MsgId* du message de rapport (ou du message de réponse) doit être défini.

#### **MQRO\_NEW\_MSG\_ID**

Il s'agit de l'action par défaut qui indique que si un rapport ou une réponse est généré suite à ce message, un nouveau *MsgId* est généré pour le message de rapport ou de réponse.

#### **MQRO\_PASS\_MSG\_ID**

Si un rapport ou une réponse est généré à la suite de ce message, le *MsgId* de ce message est copié dans le *MsgId* du message de rapport ou de réponse.

Le *MsgId* d'un message de publication sera différent pour chaque abonné qui reçoit une copie de la publication et, par conséquent, le *MsgId* copié dans le message de rapport ou de réponse sera différent pour chaque abonné.

Si cette option n'est pas spécifiée, *MQRO\_NEW\_MSG\_ID* est utilisé.

**Options d'identificateur de corrélation:** spécifiez l'une des options répertoriées pour contrôler la manière dont le *CorrelId* du message de rapport (ou du message de réponse) doit être défini.

#### **MQRO\_COPY\_MSG\_ID\_TO\_CORREL\_ID**

Il s'agit de l'action par défaut, qui indique que si un rapport ou une réponse est généré suite à ce message, le *MsgId* de ce message est copié dans le *CorrelId* du message de rapport ou de réponse.

Le *MsgId* d'un message de publication sera différent pour chaque abonné qui reçoit une copie de la publication et, par conséquent, le *MsgId* copié dans le *CorrelId* du message de rapport ou de réponse sera différent pour chacun d'eux.

#### **MQRO\_PASS\_CORREL\_ID**

Si un rapport ou une réponse est généré à la suite de ce message, le *CorrelId* de ce message est copié dans le *CorrelId* du message de rapport ou de réponse.

Le *CorrelId* d'un message de publication est spécifique à un abonné, sauf s'il utilise l'option *MQSO\_SET\_CORREL\_ID* et définit la zone d'ID SubCorrel dans le MQSD sur MQCI\_NONE. Par conséquent, il est possible que le *CorrelId* copié dans le *CorrelId* du message de rapport ou de réponse soit différent pour chacun d'eux.

Si cette option n'est pas spécifiée, *MQRO\_COPY\_MSG\_ID\_TO\_CORREL\_ID* est utilisé.

Les serveurs qui répondent aux demandes ou qui génèrent des messages de rapport doivent vérifier si les options *MQRO\_PASS\_MSG\_ID* ou *MQRO\_PASS\_CORREL\_ID* ont été définies dans le message d'origine. Si tel est le cas, les serveurs doivent effectuer l'action décrite pour ces options. Si aucun n'est défini, les serveurs doivent effectuer l'action par défaut correspondante.

**Options de disposition:** Indiquez l'une des options répertoriées pour contrôler la disposition du message d'origine lorsqu'il ne peut pas être distribué à la file d'attente de destination. L'application peut définir les options d'élimination indépendamment de la demande de rapports d'exception.

#### **MQRO\_DEAD\_LETTER\_Q**

Il s'agit de l'action par défaut qui place le message dans la file d'attente de rebut si le message ne peut pas être distribué à la file d'attente de destination. Cela se produit dans les situations suivantes:

- Lorsque l'application qui a inséré le message d'origine ne peut pas être informée de manière synchrone du problème à l'aide du code anomalie renvoyé par l'appel MQPUT ou MQPUT1. Un message de rapport d'exception est généré si l'expéditeur en a demandé un.
- Lorsque l'application qui a inséré le message d'origine a inséré une rubrique

## **MQRO\_DISCARD\_MSG**

Le message est supprimé s'il ne peut pas être distribué à la file d'attente de destination. Cela se produit dans les situations suivantes:

- Lorsque l'application qui a inséré le message d'origine ne peut pas être informée de manière synchrone du problème à l'aide du code anomalie renvoyé par l'appel MQPUT ou MQPUT1 . Un message de rapport d'exception est généré si l'expéditeur en a demandé un.
- Lorsque l'application qui a inséré le message d'origine a inséré une rubrique

Si vous souhaitez renvoyer le message d'origine à l'expéditeur, sans que le message d'origine soit placé dans la file d'attente de rebut, l'expéditeur doit spécifier MQRO\_DISCARD\_MSG avec MQRO\_EXCEPTION\_WITH\_FULL\_DATA.

## **MQRO\_PASS\_DISCARD\_ET\_EXPIRATION**

Si cette option est définie sur un message et qu'un rapport ou une réponse est généré en raison de celui-ci, le descripteur de message du rapport hérite:

- MQRO\_DISCARD\_MSG, s'il a été défini.
- Heure d'expiration restante du message (s'il ne s'agit pas d'un rapport d'expiration). S'il s'agit d'un rapport d'expiration, le délai d'expiration est de 60 secondes.

## **Option d'activité**

### **ACTIVITE MQRO**

L'utilisation de cette valeur permet de tracer la route de **tout** message sur un réseau de gestionnaires de files d'attente. L'option de rapport peut être spécifiée sur n'importe quel message utilisateur en cours, ce qui vous permet instantanément de commencer à calculer la route du message via le réseau.

Si l'application qui génère le message ne peut pas activer la génération de rapports d'activité, la génération de rapports d'activité peut être activée à l'aide d'un exit de croisement d'API fourni par les administrateurs du gestionnaire de files d'attente.

### **Remarque :**

1. Moins les gestionnaires de files d'attente du réseau sont en mesure de générer des rapports d'activité, moins la route est détaillée.
2. Les rapports d'activité peuvent être difficiles à placer dans le bon ordre pour déterminer la route empruntée.
3. Il se peut que les rapports d'activité ne parviennent pas à trouver une route vers leur destination demandée.
4. Les messages avec ce jeu d'options de rapport doivent être acceptés par n'importe quel gestionnaire de files d'attente, même s'ils ne comprennent pas l'option. Cela permet de définir l'option de rapport sur n'importe quel message utilisateur, même s'ils sont traités par un gestionnaire de files d'attente nonIBM WebSphere MQ 6.0 ou ultérieur.
5. Si un processus, qu'il s'agisse d'un gestionnaire de files d'attente ou d'un processus utilisateur, effectue une activité sur un message avec cette option définie, il peut choisir de générer et d'insérer un rapport d'activité.

**Option par défaut:** spécifiez ce qui suit si aucune option de rapport n'est requise:

### **MQRO\_AUCUN**

Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. MQRO\_NONE est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette option soit utilisée avec une autre option, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

### **Informations générales :**

1. Tous les types de rapport requis doivent être spécifiquement demandés par l'application qui envoie le message d'origine. Par exemple, si un rapport COA est demandé mais qu'un rapport d'exception n'est pas, un rapport COA est généré lorsque le message est placé dans la file d'attente de destination, mais aucun rapport d'exception n'est généré si la file d'attente de destination est saturée lorsque le

message y arrive. Si aucune option *Report* n'est définie, aucun message de rapport n'est généré par le gestionnaire de files d'attente ou l'agent MCA.

Certaines options de rapport peuvent être spécifiées même si le gestionnaire de files d'attente local ne les reconnaît pas ; cela est utile lorsque l'option doit être traitée par le gestionnaire de files d'attente de destination . Pour plus d'informations, voir «Options de rapport et indicateurs de message», à la page 947.

Si un message de rapport est demandé, le nom de la file d'attente à laquelle envoyer le rapport doit être spécifié dans la zone *ReplyToQ* . Lorsqu'un message de rapport est reçu, la nature du rapport peut être déterminée en examinant la zone *Feedback* dans le descripteur de message.

2. Si le gestionnaire de files d'attente ou l'agent MCA qui génère un message de rapport ne peut pas placer le message de rapport dans la file d'attente de réponses (par exemple, parce que la file d'attente de réponses ou la file d'attente de transmission est saturée), le message de rapport est placé dans la file d'attente de rebut. Si cette également échoue ou qu'il n'existe pas de file d'attente de rebut, l'action effectuée dépend du type de message de rapport:
  - Si le message de rapport est un rapport d'exception, le message qui a généré le rapport d'exception est laissé dans sa file d'attente de transmission, ce qui garantit que le message n'est pas perdu.
  - Pour tous les autres types de rapport, le message de rapport est supprimé et le traitement se poursuit normalement. Cela est dû au fait que le message d'origine a déjà été distribué en toute sécurité (pour les messages de rapport COA ou COD) ou qu'il n'a plus d'intérêt (pour un message de rapport d'expiration).

Une fois qu'un message de rapport a été correctement placé dans une file d'attente (file de destination ou file de transmission intermédiaire), le message n'est plus soumis à un traitement spécial ; il est traité comme tout autre message.

3. Lorsque le rapport est généré, la file d'attente *ReplyToQ* est ouverte et le message de rapport est inséré à l'aide des droits de l'utilisateur *UserIdentifier* dans le MQMD du message à l'origine du rapport, sauf dans les cas suivants:
  - Les rapports d'exception générés par un agent MCA récepteur sont placés avec les droits utilisés par l'agent MCA lorsqu'il a tenté d'insérer le message à l'origine du rapport.
  - Les rapports COA générés par le gestionnaire de files d'attente sont insérés avec les droits d'accès utilisés lorsque le message à l'origine du rapport a été inséré dans le gestionnaire de files d'attente qui génère le rapport. Par exemple, si le message a été inséré par un agent MCA récepteur à l'aide de l'identificateur utilisateur de l'agent MCA, le gestionnaire de files d'attente insère le rapport COA à l'aide de l'identificateur utilisateur de l'agent MCA.

Les applications qui génèrent des rapports doivent utiliser les mêmes droits qu'elles utilisent pour générer une réponse ; il s'agit généralement des droits de l'ID utilisateur dans le message d'origine.

Si le rapport doit être acheminé vers une destination éloignée, les expéditeurs et les destinataires peuvent décider de l'accepter ou non, de la même manière qu'ils le font pour les autres messages.

4. Si un message de rapport contenant des données est demandé:
  - Le message de rapport est toujours généré avec la quantité de données demandée par l'expéditeur du message d'origine. Si le message d'état est trop volumineux pour la file d'attente de réponses, le traitement décrit ci-dessus se produit ; le message d'état n'est jamais tronqué pour tenir dans la file d'attente de réponses.
  - Si le *Format* du message d'origine est MQFMT\_XMIT\_Q\_HEADER, les données incluses dans le rapport n'incluent pas MQXQH. Les données de rapport commencent par le premier octet des données au-delà du MQXQH dans le message d'origine. Cela se produit que la file d'attente soit ou non une file d'attente de transmission.
5. Si un message COA, COD ou de rapport d'expiration est reçu dans la file d'attente de réponses, il est garanti que le message d'origine est arrivé, a été distribué ou a expiré, selon le cas. Toutefois, si un ou plusieurs de ces messages de rapport sont demandés et ne sont pas reçus, l'inverse ne peut pas être pris en compte, car l'un des cas suivants peut s'être produit:
  - a. Le message de rapport est suspendu car un lien est arrêté.

- b. Le message de rapport est suspendu car une condition de blocage existe dans une file d'attente de transmission intermédiaire ou dans la file d'attente de réponse (par exemple, la file d'attente est saturée ou bloquée pour les insertions).
- c. Le message de rapport se trouve dans une file d'attente de rebut.
- d. Lorsque le gestionnaire de files d'attente a tenté de générer le message de rapport, il n'a pu le placer ni dans la file d'attente appropriée, ni dans la file d'attente de rebut, de sorte que le message de rapport n'a pas pu être généré.
- e. Une défaillance du gestionnaire de files d'attente s'est produite entre l'action signalée (arrivée, distribution ou expiration) et la génération du message de rapport correspondant. (Cela ne se produit pas pour les messages de rapport COD si l'application extrait le message d'origine dans une unité de travail, car le message de rapport COD est généré dans la même unité de travail.)

Les messages de rapport d'exception peuvent être conservés de la même manière pour les raisons 1, 2 et 3 ci-dessus. Toutefois, lorsqu'un agent MCA ne peut pas générer de message de rapport d'exception (le message de rapport ne peut pas être placé dans la file d'attente de réponses ou dans la file d'attente de rebut), le message d'origine reste dans la file d'attente de transmission de l'émetteur et le canal est fermé. Cela se produit indépendamment du fait que le message de rapport devait être généré à l'extrémité émettrice ou réceptrice du canal.

6. Si le message d'origine est temporairement bloqué (ce qui entraîne la génération d'un message de rapport d'exception et l'insertion du message d'origine dans une file d'attente de rebut), mais que le blocage est effacé et qu'une application lit ensuite le message d'origine dans la file d'attente de rebut et le place à nouveau à sa destination, les situations suivantes peuvent se produire:
  - Même si un message de rapport d'exception a été généré, le message d'origine arrive finalement à sa destination.
  - Plusieurs messages de rapport d'exception sont générés pour un seul message d'origine, car le message d'origine risque de rencontrer un autre blocage ultérieurement.

#### **Signaler les messages lors de l'insertion dans une rubrique:**

1. Des rapports peuvent être générés lors de l'insertion d'un message dans une rubrique. Ce message sera envoyé à tous les abonnés à la rubrique, qui peut être zéro, un ou plusieurs. Cela doit être pris en compte lorsque vous choisissez d'utiliser les options de rapport car un grand nombre de messages de rapport peuvent être générés en conséquence.
2. Lors de l'insertion d'un message dans une rubrique, il se peut que de nombreuses files d'attente de destination reçoivent une copie du message. Si certaines de ces files d'attente de destination présentent un problème, par exemple une file d'attente saturée, la réussite de l'opération MQPUT dépend de la valeur de NPMSGDLV ou de PMSGDLV (en fonction de la persistance du message). Si le paramètre est tel que la distribution des messages à la file d'attente de destination doit aboutir (par exemple, s'il s'agit d'un message persistant envoyé à un abonné durable et que PMSGDLV est défini sur ALL ou ALLDUR), la réussite est définie comme l'un des critères suivants:
  - Insertion réussie dans la file d'attente de l'abonné
  - Utilisation de MQRO\_DEAD\_LETTER\_Q et insertion réussie dans la file d'attente de rebut si la file d'attente de l'abonné ne peut pas prendre le message
  - Utilisation de MQRO\_DISCARD\_MSG si la file d'attente de l'abonné ne peut pas prendre le message.

#### **Messages de rapport pour les segments de message:**

1. Des messages de rapport peuvent être demandés pour les messages dont la segmentation est autorisée (voir la description de l'indicateur MQMF\_SEGMENTATION\_ALLOWED). Si le gestionnaire de files d'attente juge nécessaire de segmenter le message, un message de rapport peut être généré pour chacun des segments qui rencontre ensuite la condition appropriée. Les applications doivent être préparées pour recevoir plusieurs messages de rapport pour chaque type de message de rapport demandé. Utilisez la zone *GroupId* dans le message de rapport pour corrélérer les différents rapports avec l'identificateur de groupe du message d'origine, et la zone *Feedback* identifie le type de chaque message de rapport.

2. Si MQGMO\_LOGICAL\_ORDER est utilisé pour extraire des messages de rapport pour des segments, sachez que des rapports de *différents types* peuvent être renvoyés par les appels MQGET successifs. Par exemple, si les rapports COA et COD sont demandés pour un message segmenté par le gestionnaire de files d'attente, les appels MQGET des messages de rapport peuvent renvoyer les messages de rapport COA et COD imbriqués de manière imprévisible. Evitez cela en utilisant l'option MQGMO\_COMPLETE\_MSG (éventuellement avec MQGMO\_ACCEPT\_TRUNCATED\_MSG). MQGMO\_COMPLETE\_MSG entraîne le gestionnaire de files d'attente à réassembler les messages de rapport ayant le même type de rapport. Par exemple, le premier appel MQGET peut réassembler tous les messages COA relatifs au message d'origine et le deuxième appel MQGET peut réassembler tous les messages COD. Ce qui est réassemblé en premier dépend du type de message de rapport qui apparaît en premier dans la file d'attente.
3. Les applications qui elles-mêmes placent des segments peuvent spécifier des options de rapport différentes pour chaque segment. Notez toutefois les points suivants:
  - Si les segments sont extraits à l'aide de l'option MQGMO\_COMPLETE\_MSG, seules les options de rapport du *premier* segment sont prises en compte par le gestionnaire de files d'attente.
  - Si les segments sont extraits un par un et que la plupart d'entre eux possèdent l'une des options MQRO\_COD\_\*, mais pas au moins un segment, vous ne pouvez pas utiliser l'option MQGMO\_COMPLETE\_MSG pour extraire les messages de rapport avec un seul appel MQGET ou utiliser l'option MQGMO\_ALL\_SEGMENTS\_AVAILABLE pour détecter l'arrivée de tous les messages de rapport.
4. Dans un réseau MQ, les gestionnaires de files d'attente peuvent avoir des fonctions différentes. Si un message de rapport pour un segment est généré par un gestionnaire de files d'attente ou un agent MCA qui ne prend pas en charge la segmentation, le gestionnaire de files d'attente ou l'agent MCA n'inclut pas par défaut les informations de segment nécessaires dans le message de rapport, ce qui peut rendre difficile l'identification du message d'origine à l'origine de la génération du rapport. Evitez cette difficulté en demandant des données avec le message de rapport, c'est-à-dire en spécifiant les options MQRO\_\*\_WITH\_DATA ou MQRO\_\*\_WITH\_FULL\_DATA appropriées. Toutefois, sachez que si MQRO\_\*\_WITH\_DATA est spécifié, *moins de* 100 octets de données de message d'application peuvent être renvoyés à l'application qui extrait le message de rapport, si le message de rapport est généré par un gestionnaire de files d'attente ou un agent MCA qui ne prend pas en charge la segmentation.

**Contenu du descripteur de message d'un message de rapport:** Lorsque le gestionnaire de files d'attente ou l'agent MCA génère un message de rapport, il définit les zones du descripteur de message sur les valeurs suivantes, puis place le message de manière normale.

Tableau 501. Valeurs utilisées pour les zones MQMD lorsqu'un message de rapport est généré par le système

Zone dans MQMD	Valeur utilisée
<i>StrucId</i>	ID_STRUCD_MQM
<i>Version</i>	MQMD_VERSION_2
<i>Report</i>	MQRO_AUCUN
<i>MsgType</i>	MQMT_REPORT
<i>Expiry</i>	MQEI_UNLIMITED
<i>Feedback</i>	Selon la nature du rapport (MQFB_COA, MQFB_COD, MQFB_EXPIRATION ou une valeur MQRC_*)
<i>Encoding</i>	Copié à partir du descripteur de message d'origine
<i>CodedCharSetId</i>	Copié à partir du descripteur de message d'origine
<i>Format</i>	Copié à partir du descripteur de message d'origine
<i>Priority</i>	Copié à partir du descripteur de message d'origine
<i>Persistence</i>	Copié à partir du descripteur de message d'origine

Tableau 501. Valeurs utilisées pour les zones MQMD lorsqu'un message de rapport est généré par le système (suite)

Zone dans MQMD	Valeur utilisée
<i>MsgId</i>	Comme indiqué par les options de rapport dans le descripteur de message d'origine
<i>CorrelId</i>	Comme indiqué par les options de rapport dans le descripteur de message d'origine
<i>BackoutCount</i>	0
<i>ReplyToQ</i>	Espaces vides
<i>ReplyToQMgr</i>	Nom du gestionnaire de files d'attente
<i>UserIdentifier</i>	Comme défini par l'option MQPMO_PASS_IDENTITY_CONTEXT
<i>AccountingToken</i>	Comme défini par l'option MQPMO_PASS_IDENTITY_CONTEXT
<i>ApplIdentityData</i>	Comme défini par l'option MQPMO_PASS_IDENTITY_CONTEXT
<i>PutApplType</i>	MQAT_QMGR, ou selon le cas, pour l'agent MCA
<i>PutApplName</i>	28 premiers octets du nom du gestionnaire de files d'attente ou du nom de l'agent MCA. Pour les messages de rapport générés par le pont IMS, cette zone contient le nom de groupe XCF et le nom de membre XCF du système IMS auquel le message est lié.
<i>PutDate</i>	Date d'envoi du message de rapport
<i>PutTime</i>	Heure d'envoi du message de rapport
<i>ApplOriginData</i>	Espaces vides
<i>GroupId</i>	Copié à partir du descripteur de message d'origine
<i>MsgSeqNumber</i>	Copié à partir du descripteur de message d'origine
<i>Offset</i>	Copié à partir du descripteur de message d'origine
<i>MsgFlags</i>	Copié à partir du descripteur de message d'origine
<i>OriginalLength</i>	Copié à partir du descripteur de message d'origine s'il n'est pas défini par MQOL_UNDEFINED, et défini sur la longueur des données de message d'origine dans le cas contraire

Une application générant un rapport est recommandée pour définir des valeurs similaires, à l'exception des suivantes:

- La zone *ReplyToQMgr* peut être mise à blanc (le gestionnaire de files d'attente le remplace par le nom du gestionnaire de files d'attente local lors de l'insertion du message).
- Définissez les zones de contexte à l'aide de l'option qui aurait été utilisée pour une réponse, normalement MQPMO\_PASS\_IDENTITY\_CONTEXT.

**Analyse de la zone de rapport:** la zone *Report* contient des sous-zones ; pour cette raison, les applications qui doivent vérifier si l'expéditeur du message a demandé un rapport particulier doivent utiliser l'une des techniques décrites dans «Analyse de la zone de rapport», à la page 949.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1. La valeur initiale de cette zone est MQRO\_NONE.

### **MsgType (MQLONG) pour MQMD**

Indique le type du message. Les types de message sont regroupés comme suit:

#### **MQMT\_SYSTEM\_FIRST**

Valeur la plus basse pour les types de message définis par le système.

## **MQMT\_SYSTEM\_LAST**

Valeur la plus élevée pour les types de message définis par le système.

Les valeurs suivantes sont actuellement définies dans la plage système:

## **MQMT\_DATAGRAM**

Le message ne requiert pas de réponse.

## **MQMT\_REQUEST**

Le message requiert une réponse.

Indiquez le nom de la file d'attente à laquelle envoyer la réponse dans la zone *ReplyToQ*. La zone *Report* indique comment définir les valeurs *MsgId* et *CorrelId* de la réponse.

## **MQMT\_REPLY**

Le message correspond à la réponse à un message de demande antérieur (MQMT\_REQUEST). Le message doit être envoyé à la file d'attente indiquée par la zone *ReplyToQ* du message de demande. Utilisez la zone *Report* de la demande pour contrôler comment définir les valeurs *MsgId* et *CorrelId* de la réponse.

**Remarque :** Le gestionnaire de files d'attente n'applique pas la relation de demande / réponse ; il s'agit d'une responsabilité d'application.

## **MQMT\_REPORT**

Le message signale une occurrence attendue ou inattendue, généralement liée à un autre message (par exemple, un message de demande contenant des données non valides a été reçu). Envoyez le message à la file d'attente indiquée par la zone *ReplyToQ* du descripteur de message du message d'origine. Définissez les zones *Feedback* pour indiquer la nature du rapport. Utilisez la zone *Report* du message d'origine pour contrôler comment définir les valeurs *MsgId* et *CorrelId* du message de rapport.

Les messages de rapport générés par le gestionnaire de files d'attente ou l'agent MCA sont toujours envoyés à la file d'attente *ReplyToQ*, avec les zones *Feedback* et *CorrelId* définies comme indiqué ci-dessus.

Les valeurs définies par l'application peuvent également être utilisées. Ils doivent être compris dans la plage suivante:

## **MQMT\_APPL\_FIRST**

Valeur la plus faible pour les types de message définis par l'application.

## **MQMT\_APPL\_LAST**

Valeur la plus élevée pour les types de message définis par l'application.

Pour les appels MQPUT et MQPUT1, la valeur *MsgType* doit être comprise dans la plage définie par le système ou dans la plage définie par l'application ; si ce n'est pas le cas, l'appel échoue avec le code anomalie MQRC\_MSG\_TYPE\_ERROR.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1. La valeur initiale de cette zone est MQMT\_DATAGRAM.

## **Expiration (MQLONG) pour MQMD**

Il s'agit d'une période de temps exprimée en dixièmes de seconde, définie par l'application qui insère le message. Le message peut être supprimé s'il n'a pas été supprimé de la file d'attente de destination avant l'expiration de ce délai.

Par exemple, pour définir une minute pour l'heure d'expiration, vous devez définir **MQMD.Expiry** à 600.

La valeur est décrémentée pour refléter le temps passé par le message dans la file d'attente de destination, ainsi que dans les files d'attente de transmission intermédiaires si l'insertion est dans une file d'attente éloignée. Il peut également être décrémenté par les agents de canal de message pour refléter les temps de transmission, si ceux-ci sont significatifs. De même, une application qui transmet ce message à une autre file d'attente peut décrémenter la valeur si nécessaire, si elle a conservé le message pendant un temps significatif. Toutefois, le délai d'expiration est traité comme approximatif et la valeur n'a pas besoin d'être décrémentée pour refléter de petits intervalles de temps.

Lorsque le message est extrait par une application à l'aide de l'appel MQGET, la zone *Expiry* représente l'heure d'expiration qui reste.

Une fois le délai d'expiration d'un message écoulé, il peut être supprimé par le gestionnaire de files d'attente. Le message est supprimé lorsqu'un appel MQGET de consultation ou de non-consultation se produit et qu'il aurait renvoyé le message s'il n'était pas déjà arrivé à expiration. Par exemple, un appel MQGET de non-consultation avec la zone *MatchOptions* de MQGMO définie sur MQMO\_NONE en lecture à partir d'une file d'attente commandée FIFO supprime tous les messages arrivés à expiration jusqu'au premier message non arrivé à expiration. Avec une file d'attente ordonnée de priorité, le même appel va supprimer les messages arrivés à expiration de priorité supérieure et les messages de priorité égale qui sont arrivés dans la file d'attente avant le premier message non arrivé à expiration.

Un message arrivé à expiration n'est jamais renvoyé à une application (par un appel MQGET de recherche ou de non-recherche). Par conséquent, la valeur de la zone *Expiry* du descripteur de message après un appel MQGET réussi est supérieure à zéro ou la valeur spéciale MQEI\_UNLIMITED.

Si un message est inséré dans une file d'attente éloignée, il peut expirer (et être supprimé) alors qu'il se trouve dans une file d'attente de transmission intermédiaire, avant que le message n'atteigne la file d'attente de destination.

Un rapport est généré lorsqu'un message arrivé à expiration est supprimé, si le message a spécifié l'une des options de rapport MQRO\_EXPIRATION\_\*. Si aucune de ces options n'est spécifiée, aucun rapport de ce type n'est généré ; le message est supposé ne plus être pertinent après cette période (peut-être parce qu'un message ultérieur l'a remplacé).

Pour un message inséré dans un point de synchronisation, l'intervalle d'expiration commence au moment où le message est inséré et non au moment où le point de synchronisation est validé. Il est possible que l'intervalle d'expiration se soit écoulé avant que le point de synchronisation ne soit validé. Dans ce cas, le message sera supprimé à un moment donné après l'opération de validation et le message ne sera pas renvoyé à une application en réponse à une opération MQGET.

Tout autre programme qui supprime des messages en fonction de l'heure d'expiration doit également envoyer un message de rapport approprié si un tel message a été demandé.

#### Remarques :

1. Si un message est inséré avec une heure *Expiry* égale à zéro ou un nombre supérieur à 999 999 999, l'appel MQPUT ou MQPUT1 échoue avec le code anomalie MQRC\_EXPIRY\_ERROR; aucun message de rapport n'est généré dans ce cas.

Pour activer le code anomalie 2013, MQRC\_EXPIRY\_ERROR, vous devez activer la variable d'environnement AMQ\_ENFORCE\_MAX\_EXPIRY\_ERROR.

Voici un exemple d'utilisation de Linux:

```
$ export AMQ_ENFORCE_MAX_EXPIRY_ERROR=True
```

Notez que:

- Il est important d'exporter la variable
  - La valeur réelle est ignorée, mais l'utilisation de True peut être utile lors de la révision de la configuration.
2. Etant donné qu'un message dont le délai d'expiration est écoulé peut ne pas être supprimé avant une date ultérieure, il se peut que des messages d'une file d'attente aient dépassé leur délai d'expiration et qu'ils ne puissent donc pas être extraits. Ces messages sont néanmoins pris en compte dans le nombre de messages dans la file d'attente à toutes fins, y compris le déclenchement de la profondeur.  
  
Si un abonné / consommateur (client) tente d'obtenir un message et que ce message est arrivé à expiration, le client ne reçoit rien car le message a été supprimé car il était trop ancien. En outre, le client ne recevra aucun message d'erreur.
  3. Un rapport d'expiration est généré, s'il est demandé, lorsque le message est supprimé et non lorsqu'il devient éligible à la suppression.



4. La suppression d'un message arrivé à expiration et la génération d'un rapport d'expiration, le cas échéant, ne font jamais partie de l'unité d'oeuvre de l'application, même si la suppression du message a été planifiée suite à un appel MQGET effectué dans une unité d'oeuvre.
5. Si un message presque arrivé à expiration est extrait par un appel MQGET au sein d'une unité de travail et que l'unité de travail est ensuite annulée, le message peut être supprimé avant de pouvoir être à nouveau extrait.
6. Si un message presque arrivé à expiration est verrouillé par un appel MQGET avec MQGMO\_LOCK, le message peut être supprimé avant de pouvoir être extrait par un appel MQGET avec MQGMO\_MSG\_UNDER\_CURSOR; le code anomalie MQRC\_NO\_MSG\_UNDER\_CURSOR est renvoyé sur cet appel MQGET suivant si cela se produit.
7. Lorsqu'un message de demande dont le délai d'expiration est supérieur à zéro est extrait, l'application peut effectuer l'une des actions suivantes lorsqu'elle envoie le message de réponse:
  - Copiez le délai d'expiration restant entre le message de demande et le message de réponse.
  - Définissez l'heure d'expiration dans le message de réponse sur une valeur explicite supérieure à zéro.
  - Définissez le délai d'expiration dans le message de réponse sur MQEI\_UNLIMITED.

L'action à effectuer dépend de la conception de l'application. Toutefois, l'action par défaut pour placer des messages dans une file d'attente de messages non livrés doit être de conserver le délai d'expiration restant du message et de continuer à le décrémenter.

8. Les messages de déclenchement sont toujours générés avec MQEI\_UNLIMITED.
9. Un message (généralement dans une file d'attente de transmission) dont le nom *Format* est MQFMT\_XMIT\_Q\_HEADER comporte un deuxième descripteur de message dans MQXQH. Il est donc associé à deux zones *Expiry*. Dans ce cas, il convient de noter les points supplémentaires suivants:
  - Lorsqu'une application place un message dans une file d'attente éloignée, le gestionnaire de files d'attente place initialement le message dans une file d'attente de transmission locale et préfixe les données de message d'application avec une structure MQXQH. Le gestionnaire de files d'attente définit les valeurs des deux zones *Expiry* de sorte qu'elles soient identiques à celles spécifiées par l'application.  
  
Si une application insère un message directement dans une file d'attente de transmission locale, les données de message doivent déjà commencer par une structure MQXQH et le nom de format doit être MQFMT\_XMIT\_Q\_HEADER. Dans ce cas, l'application n'a pas besoin de définir les valeurs de ces deux zones *Expiry* pour qu'elles soient identiques. (Le gestionnaire de files d'attente vérifie que la zone *Expiry* dans MQXQH contient une valeur valide et que les données de message sont suffisamment longues pour l'inclure). Pour une application qui peut écrire directement dans la file d'attente de transmission, l'application doit créer un en-tête de file d'attente de transmission avec le descripteur de message intégré. Toutefois, si la valeur d'expiration du descripteur de message écrit dans la file d'attente de transmission est incohérente avec la valeur du descripteur de message intégré, une erreur d'expiration est rejetée.
  - Lorsqu'un message portant le nom *Format* MQFMT\_XMIT\_Q\_HEADER est extrait d'une file d'attente (qu'il s'agisse d'une file d'attente normale ou de transmission), le gestionnaire de files d'attente décrémente ces deux zones *Expiry* avec le temps d'attente de la file d'attente. Aucune erreur n'est générée si les données de message ne sont pas suffisamment longues pour inclure la zone *Expiry* dans le MQXQH.
  - Le gestionnaire de files d'attente utilise la zone *Expiry* dans le descripteur de message distinct (c'est-à-dire, pas celui du descripteur de message intégré dans la structure MQXQH) pour tester si le message peut être supprimé.
  - Si les valeurs initiales des deux zones *Expiry* sont différentes, l'heure *Expiry* dans le descripteur de message distinct lors de l'extraction du message peut être supérieure à zéro (de sorte que le message ne peut pas être supprimé), alors que l'heure indiquée dans la zone *Expiry* de MQXQH s'est écoulée. Dans ce cas, la zone *Expiry* de MQXQH est définie sur zéro.

10. Le délai d'expiration d'un message de réponse renvoyé par le pont IMS est illimité sauf si MQIIH\_PASS\_EXPIRATION est défini dans la zone Indicateurs de MQIIH. Pour plus d'informations, voir [Indicateurs](#).

La valeur spéciale suivante est reconnue:

### **MQEI\_UNLIMITED**

Le message a un délai d'expiration illimité.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1. La valeur initiale de cette zone est MQEI\_UNLIMITED.

#### *Messages arrivés à expiration sur z/OS*

Sous IBM MQ for z/OS, les messages arrivés à expiration sont supprimés par le prochain appel MQGET approprié.

Toutefois, si aucun appel de ce type n'est effectué, le message arrivé à expiration n'est pas supprimé et, pour certaines files d'attente, un grand nombre de messages arrivés à expiration peuvent s'accumuler. Pour y remédier, définissez le gestionnaire de files d'attente de sorte qu'il analyse périodiquement les files d'attente et qu'il rejette les messages arrivés à expiration dans une ou plusieurs files d'attente de l'une des manières suivantes:

#### **Recherche périodique**

Vous pouvez spécifier une période à l'aide de l'attribut de gestionnaire de files d'attente EXPRYINT (intervalle d'expiration). Chaque fois que l'intervalle d'expiration est atteint, le gestionnaire de files d'attente recherche les files d'attente candidates qui méritent d'être examinées pour supprimer les messages arrivés à expiration.

Le gestionnaire de files d'attente gère les informations sur les messages arrivés à expiration dans chaque file d'attente et sait si une analyse des messages arrivés à expiration est utile. Ainsi, seule une sélection de files d'attente est analysée à tout moment.

Les files d'attente partagées sont analysées par un seul gestionnaire de files d'attente dans un groupe de partage de files d'attente. En règle générale, il s'agit du premier gestionnaire de files d'attente à redémarrer ou du premier gestionnaire de files d'attente pour lequel EXPRYINT est défini. Si ce gestionnaire de files d'attente s'arrête, un autre gestionnaire de files d'attente du groupe de partage de files d'attente prend en charge l'analyse de la file d'attente. Définissez la valeur de l'intervalle d'expiration pour tous les gestionnaires de files d'attente d'un groupe de partage de files d'attente sur la même valeur.


Notez que le traitement d'expiration a lieu pour chaque file d'attente lorsqu'un gestionnaire de files d'attente redémarre, quel que soit le paramètre EXPRYINT.



#### **Demande explicite**

Exécutez la commande REFRESH QMGR TYPE (EXPIRATION) en spécifiant la ou les files d'attente que vous souhaitez analyser.

#### *Application des délais d'expiration inférieurs*

Les administrateurs peuvent limiter le délai d'expiration des messages placés dans une file d'attente ou une rubrique en utilisant l'attribut **CAPEXPY** spécifié dans l'attribut **CUSTOM** dans la file d'attente ou la rubrique.

**Important :**  Vous ne pouvez pas utiliser l'attribut **CAPEXPY** introduit dans IBM MQ 9.3.1 avec un cluster si le référentiel complet est sous z/OS.

  Depuis la IBM MQ 9.3.1, les administrateurs peuvent limiter le délai d'expiration à l'aide de l'attribut **CAPEXPY** d'une file d'attente ou d'une rubrique sans avoir à le spécifier dans l'attribut **CUSTOM**. Si une file d'attente ou une rubrique a déjà l'attribut **CAPEXPY** défini dans l'attribut **CUSTOM**, vous devez supprimer l'attribut **CAPEXPY** de l'attribut **CUSTOM** avant de modifier le nouvel attribut **CAPEXPY**. Vous pouvez effectuer cette opération en une seule commande, par exemple:

```
ALTER QLOCAL(Q1) CAEXPY(1000) CUSTOM('')
```

**Remarque :** Lorsque vous migrez un objet à partir d'une version antérieure du produit, votre valeur **CAPEXPRT** est définie sur la valeur par défaut NOLIMIT. Si l'attribut **CAPEXPRT** a été défini dans l'attribut **CUSTOM**, cette option est prioritaire.

Si vous souhaitez utiliser le nouvel attribut **CAPEXPRT**, vous devez d'abord supprimer l'attribut **CAPEXPRT** de l'attribut **CUSTOM**. La définition des deux options ne fonctionne pas.

**V 9.3.1** **Multi** L'attribut **CAPEXPRT** révisé défini directement sur les files d'attente ou les rubriques (pas dans l'attribut **CUSTOM**) est un attribut de cluster. Notez que toutes les instances d'une file d'attente de cluster doivent utiliser la même valeur pour leur attribut **CAPEXPRT**. Il est toujours possible qu'une file d'attente de transmission réduise le délai d'expiration d'un message si **CAPEXPRT** a été défini sur la file d'attente de transmission et que la valeur est inférieure à l'attribut **CAPEXPRT** de la file d'attente de cluster.

Une heure d'expiration spécifiée dans la zone **Expiry** du MQMD par une application, qui est supérieure à la valeur **CAPEXPRT** spécifiée dans la file d'attente ou la rubrique, sera remplacée par cette valeur **CAPEXPRT**. Une heure d'expiration spécifiée par une application, qui est inférieure à la valeur **CAPEXPRT**, sera utilisée.

Notez que la valeur de **CAPEXPRT** est exprimée en dixièmes de secondes, de sorte qu'une minute a une valeur de 600.

Si plusieurs objets sont utilisés sur le chemin de résolution, par exemple, lorsqu'un message est inséré dans un alias ou une file d'attente éloignée, la valeur la plus basse de toutes les valeurs **CAPEXPRT** est utilisée comme limite supérieure pour l'expiration des messages.

Les modifications apportées aux valeurs **CAPEXPRT** prennent effet immédiatement. La valeur d'expiration est évaluée pour chaque insertion dans une file d'attente ou une rubrique et est donc sensible à la résolution d'objet, qui peut différer entre chaque opération d'insertion.

Toutefois, notez que les messages existants dans la file d'attente, avant une modification dans **CAPEXPRT**, ne sont pas affectés par la modification (c'est-à-dire que leur heure d'expiration reste intacte). Seuls les nouveaux messages placés dans la file d'attente après la modification dans **CAPEXPRT** ont la nouvelle heure d'expiration.

Dans un cluster où une insertion est effectuée dans une file d'attente ouverte avec MQOO\_BIND\_NOT\_FIXED, des valeurs d'expiration différentes peuvent être affectées aux messages à chaque insertion, en fonction de la valeur **CAPEXPRT** définie pour la file d'attente de transmission, utilisée par le canal, qui envoie le message au gestionnaire de files d'attente cible sélectionné.

Notez qu'une insertion dans une file d'attente ou une rubrique par une application JMS spécifiant un délai de distribution échoue avec MQRC\_EXPIRY\_ERROR, si le délai de distribution est supérieur au délai d'expiration résolu pour la file d'attente ou la rubrique cible. Un attribut **CAPEXPRT** défini sur une file d'attente résolue pour une destination JMS peut provoquer cette erreur.

**Remarque :** **CAPEXPRT** ne doit pas être utilisé sur les files d'attente qui contiennent des messages IBM MQ générés en interne, tels que SYSTEM.CLUSTER.\* et SYSTEM.PROTECTION.POLICY.QUEUE.

#### Référence associée

[DEFINE QUEUES](#)

[DEFINE TOPIC](#)

#### Commentaires en retour (MQLONG) pour MQMD

La zone Feedback est utilisée avec un message de type MQMT\_REPORT pour indiquer la nature du rapport et n'est significative qu'avec ce type de message.

La zone peut contenir l'une des valeurs MQFB\_\* ou l'une des valeurs MQRC\_\*. Les codes retour sont regroupés comme suit:

#### MQFB\_AUCUN

Aucun commentaire n'a été fourni.

#### MQFB\_SYSTEM\_FIRST

Valeur la plus faible pour les commentaires générés par le système.

## **MQFB\_SYSTEM\_LAST**

Valeur la plus élevée pour les commentaires générés par le système.

La plage des codes retour générés par le système MQFB\_SYSTEM\_FIRST à MQFB\_SYSTEM\_LAST inclut les codes retour généraux répertoriés dans cette rubrique (MQFB\_\*), ainsi que les codes raison (MQRC\_\*) qui peuvent se produire lorsque le message ne peut pas être inséré dans la file d'attente de destination.

## **MQFB\_APPL\_FIRST**

Valeur la plus faible pour les commentaires générés par l'application.

## **MQFB\_APPL\_LAST**

Valeur la plus élevée pour les commentaires générés par l'application.

Les applications qui génèrent des messages de rapport ne doivent pas utiliser de codes retour dans la plage système (autre que MQFB\_QUIT), sauf si elles souhaitent simuler des messages de rapport générés par le gestionnaire de files d'attente ou l'agent MCA.

Dans les appels MQPUT ou MQPUT1, la valeur spécifiée doit être MQFB\_NONE ou être comprise dans la plage système ou la plage d'applications. Cette case est cochée quelle que soit la valeur de *MsgType*.

### **Codes retour généraux:**

#### **MQFB\_COA**

Confirmation de l'arrivée dans la file d'attente de destination (voir MQRO\_COA).

#### **MQFB\_COD**

Confirmation de la livraison à l'application réceptrice (voir MQRO\_COD).

#### **EXPIRATION\_MQFB**

Le message a été supprimé car il n'avait pas été supprimé de la file d'attente de destination avant que son délai d'expiration ne soit écoulé.

#### **MQFB\_PAN**

Notification d'action positive (voir MQRO\_PAN).

#### **MQFB\_NAN**

Notification d'action négative (voir MQRO\_NAN).

#### **MQFB\_QUIT**

Arrêtez l'application.

Cela peut être utilisé par un programme de planification de la charge de travail pour contrôler le nombre d'instances d'un programme d'application en cours d'exécution. L'envoi d'un message MQMT\_REPORT avec ce code retour à une instance du programme d'application indique à cette instance qu'elle doit arrêter le traitement. Toutefois, le respect de cette convention est du ressort de l'application ; elle n'est pas appliquée par le gestionnaire de files d'attente.

### **Codes de retour d'informations du canal:**

#### **MQFB\_CHANNEL\_COMPLETED**

Un canal s'est arrêté normalement.

#### **MQFB\_CHANNEL\_FAIL**

Un canal s'est arrêté de manière anormale et passe à l'état ARRETE.

#### **MQFB\_CHANNEL\_FAIL\_RETRY**

Un canal s'est arrêté de manière anormale et passe à l'état RETRY.

### **IMS-bridge feedback codes**

Ces codes sont utilisés lorsqu'un code de détection IMS-OTMA inattendu est reçu. Le code de détection ou, lorsque le code de détection est 0x1A, le code raison associé à ce code de détection, est indiqué dans le *commentaire en retour*.

1. Pour les codes *Feedback* compris entre MQFB\_IMS\_FIRST (300) et MQFB\_IMS\_LAST (399), un code de détection autre que 0x1A a été reçu. Le *code de détection* est donné par l'expression (*Feedback* - MQFB\_IMS\_FIRST+1)

2. Pour les codes *Feedback* dans la plage MQFB\_IMS\_NACK\_1A\_REASON\_FIRST (600) à MQFB\_IMS\_NACK\_1A\_REASON\_LAST (855), un code de détection 0x1A a été reçu. Le *code anomalie* associé au code d'analyse est donné par l'expression (*Feedback* - MQFB\_IMS\_NACK\_1A\_REASON\_FIRST)

La signification des codes de détection IMS-OTMA et des codes anomalie correspondants est décrite dans le document *Open Transaction Manager Access Guide and Reference*.

Les codes retour suivants peuvent être générés par le pont IMS :

**MQFB\_DATA\_LENGTH\_ZERO**

La longueur d'un segment était égale à zéro dans les données d'application du message.

**MQFB\_DONNÉES\_LONGUEUR\_NÉGATIVE**

Une longueur de segment était négative dans les données d'application du message.

**MQFB\_DATA\_LENGTH\_TOO\_BIG**

Une longueur de segment était trop grande dans les données d'application du message.

**MQFB\_BUFFER\_OVERFLOW**

La valeur de l'une des zones de longueur entraînerait le dépassement de la mémoire tampon de messages par les données.

**MQFB\_LENGTH\_OFF\_BY\_ONE**

La valeur de l'une des zones de longueur était trop courte de 1 octet.

**MQFB\_IIH\_ERREUR**

La zone *Format* dans MQMD indique MQFMT\_IMS, mais le message ne commence pas par une structure MQIIH valide.

**MQFB\_NOT\_AUTHORIZED\_FOR\_IMS**

L'ID utilisateur contenu dans le descripteur de message MQMD ou le mot de passe contenu dans la zone *Authenticator* de la structure MQIIH n'a pas pu être validé par le pont IMS . Par conséquent, le message n'a pas été transmis à IMS.

**MQFB\_IMS\_ERREUR**

Une erreur inattendue a été renvoyée par IMS. Consultez le journal des erreurs IBM MQ sur le système sur lequel réside le pont IMS pour plus d'informations sur l'erreur.

**MQFB\_IMS\_FIRST**

Lorsque le code de détection IMS-OTMA n'est pas 0x1A, les codes retour générés par IMS sont compris entre MQFB\_IMS\_FIRST (300) et MQFB\_IMS\_LAST (399). Le code de détection IMS-OTMA lui-même est *Feedback* moins MQFB\_IMS\_ERROR.

**MQFB\_IMS\_LAST**

Valeur la plus élevée pour les commentaires générés par IMS lorsque le code de détection n'est pas 0x1A.

**MQFB\_IMS\_NACK\_1A\_REASON\_FIRST**

Lorsque le code de détection est 0x1A, les codes retour générés par IMS sont compris entre MQFB\_IMS\_NACK\_1A\_REASON\_FIRST (600) et MQFB\_IMS\_NACK\_1A\_REASON\_LAST (855).

**MQFB\_IMS\_NACK\_1A\_REASON\_LAST**

Valeur la plus élevée pour les commentaires générés par IMS lorsque le code de détection est 0x1A

**CICS-bridge feedback codes:** les codes retour suivants peuvent être générés par CICS bridge:

**MQFB\_CICS\_APPL\_ABENDED**

Le programme d'application indiqué dans le message s'est arrêté de manière anormale. Ce code retour apparaît uniquement dans la zone *Reason* de la structure MQDLH.

**MQFB\_CICS\_APPL\_NOT\_STARTED**

La commande EXEC CICS LINK pour le programme d'application spécifié dans le message a échoué. Ce code retour apparaît uniquement dans la zone *Reason* de la structure MQDLH.

**MQFB\_CICS\_BRIDGE\_FAILURE**

CICS bridge s'est arrêté de manière anormale sans avoir terminé le traitement d'erreur normal.

**MQFB\_CICS\_CCSID\_ERREUR**

Identificateur de jeu de caractères incorrect.

**MQFB\_CICS\_CIH\_ERREUR**

Structure d'en-tête d'informations CICS manquante ou non valide.

**MQFB\_CICS\_COMMAREA\_ERREUR**

Longueur de la zone de communication CICS incorrecte.

**MQFB\_CICS\_CORREL\_ID\_ERREUR**

Identificateur de corrélation incorrect.

**MQFB\_CICS\_DLQ\_ERREUR**

La tâche CICS bridge n'a pas pu copier une réponse à cette demande dans la file d'attente de rebut. La demande a été annulée.

**MQFB\_CICS\_ENCODING\_ERROR**

Codage non valide.

**MQFB\_CICS\_INTERNAL\_ERROR**

CICS bridge a rencontré une erreur inattendue.

Ce code retour apparaît uniquement dans la zone *Reason* de la structure MQDLH.

**MQFB\_CICS\_NOT\_AUTHORIZED**

ID utilisateur non autorisé ou mot de passe incorrect.

Ce code retour apparaît uniquement dans la zone *Reason* de la structure MQDLH.

**MQFB\_CICS\_UOW\_BACKED\_OUT**

L'unité de travail a été annulée pour l'une des raisons suivantes:

- Une erreur a été détectée lors du traitement d'une autre demande dans la même unité d'oeuvre.
- Une fin anormale CICS s'est produite alors que l'unité de travail était en cours.

**MQFB\_ERREUR UOW\_CICS\_**

Zone de contrôle d'unité de travail *UOWControl* incorrecte.

**Codes de retour de message de routage de trace:****ACTIVITE MQFB\_MQ**

Utilisé avec le format MQFMT\_EMBEDDED\_PCF pour autoriser l'option des données utilisateur suivant les rapports d'activité.

**ACTIVITÉS MQFB\_MAX\_ACTIVITÉS**

Renvoyé lorsque le message de trace-route est supprimé car le nombre d'activités dans lesquelles le message a été impliqué dépasse la limite du nombre maximal d'activités.

**MQFB\_NON RÉACHEMINEMENT**

Renvoyé lorsque le message de trace-route est supprimé car il est sur le point d'être envoyé à un gestionnaire de files d'attente éloignées qui ne prend pas en charge les messages de trace-route.

**MQFB\_NOT\_DELIVERED**

Renvoyé lorsque le message de trace-route est supprimé car il est sur le point d'être placé dans une file d'attente locale.

**MQFB\_UNUPPORT\_FORWARDING**

Renvoyé lorsque le message de trace-route est supprimé car une valeur du paramètre de réacheminement n'est pas reconnue et se trouve dans le masque de bits rejeté.

**MQFB\_UNSUPPORTED\_DISTRIBUTION**

Renvoyé lorsque le message de trace-route est supprimé car une valeur du paramètre de distribution n'est pas reconnue et se trouve dans le masque de bits rejeté.

**Codes anomalie IBM MQ:** pour les messages de rapport d'exception, *Feedback* contient un code anomalie IBM MQ . Les codes anomalie possibles sont les suivants:

**MQRC\_PUT\_INHIBÉ**

(2051, X'803') Appels d'insertion bloqués pour la file d'attente.

**MQRC\_Q\_FULL**

(2053, X'805') La file d'attente contient déjà le nombre maximal de messages.

**MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') Non autorisé pour l'accès.

**MQRC\_Q\_ESPACE\_NON\_DISPONIBLE**

(2056, X'808') Aucun espace disponible sur le disque pour la file d'attente.

**MQRC\_PERSISTENT\_NOT\_ALLOWED**

(2048, X'800') La file d'attente ne prend pas en charge les messages persistants.

**MQRC\_MSG\_TOO\_BIG\_FOR\_Q\_MGR**

(2031, X'7EF') Longueur de message supérieure à la longueur maximale pour le gestionnaire de files d'attente.

**MQRC\_MSG\_TOO\_BIG\_FOR\_Q**

(2030, X'7EE') Longueur de message supérieure au maximum pour la file d'attente.

Pour obtenir la liste complète des codes anomalie, voir:

- Pour IBM MQ for z/OS, voir [Codes achèvement et raison d'API](#).
- Pour toutes les autres plateformes, voir [Codes achèvement et raison d'API](#).

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1. La valeur initiale de cette zone est MQFB\_NONE.

**Codage (MQLONG) pour MQMD**

Indique le codage numérique des données numériques dans le message ; il ne s'applique pas aux données numériques dans la structure MQMD elle-même. Le codage numérique définit la représentation utilisée pour les entiers binaires, les entiers décimaux condensés et les nombres à virgule flottante.

Sous z/OS, la partie entière binaire de la zone Encoding est également utilisée pour spécifier le codage d'entier des données de caractères dans le corps du message lorsque l'identificateur de jeu de caractères correspondant indique que la représentation du jeu de caractères dépend du codage utilisé pour les entiers binaires. Cela n'affecte que certains jeux de caractères multi-octets (par exemple, les jeux de caractères UTF-16).

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données. Le gestionnaire de files d'attente ne vérifie pas que la zone est valide. La valeur spéciale suivante est définie:

**MQENC\_NATIVE**

Le codage est la valeur par défaut du langage de programmation et de la machine sur lesquels l'application s'exécute.

**Remarque :** La valeur de cette constante dépend du langage de programmation et de l'environnement. Pour cette raison, les applications doivent être compilées à l'aide des fichiers d'entête, de macro, de COPY ou d'INCLUDE appropriés à l'environnement dans lequel l'application sera exécutée.

Les applications qui placent des messages spécifient généralement MQENC\_NATIVE. Les applications qui extraient des messages doivent comparer cette zone à la valeur MQENC\_NATIVE; si les valeurs diffèrent, l'application peut avoir besoin de convertir des données numériques dans le message. Utilisez l'option MQGMO\_CONVERT pour demander au gestionnaire de files d'attente de convertir le message dans le cadre du traitement de l'appel MQGET. Pour plus d'informations sur la façon dont la zone Encoding est construite, voir «Codages de machine», à la page 944.

Si vous spécifiez l'option MQGMO\_CONVERT dans l'appel MQGET, cette zone est une zone d'entrée-sortie. La valeur spécifiée par l'application est le codage vers lequel convertir les données de message si nécessaire. Si la conversion aboutit ou n'est pas nécessaire, la valeur reste inchangée. Si la conversion échoue, la valeur après l'appel MQGET représente le codage du message non converti renvoyé à l'application.

Dans d'autres cas, il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La valeur initiale de cette zone est MQENC\_NATIVE.

### **CodedCharSetId (MQLONG) pour MQMD**

Cette zone indique l'identificateur de jeu de caractères des données de type caractère dans le corps du message.

**Remarque :** Les données de type caractère dans MQMD et les autres structures de données MQ qui sont des paramètres sur les appels doivent figurer dans le jeu de caractères du gestionnaire de files d'attente. Il est défini par l'attribut **CodedCharSetId** du gestionnaire de files d'attente ; pour plus de détails sur cet attribut, voir «Attributs du gestionnaire de files d'attente», à la page 835 .

Si cette zone est définie sur MQCCSI\_Q\_MGR lors de l'appel de MQGET avec MQGMO\_CONVERT dans les options, le comportement est différent entre les applications client et serveur. Pour les applications serveur, la page de codes utilisée pour la conversion de caractères est la CodedCharSetId du gestionnaire de files d'attente ; pour les applications client, la page de codes utilisée pour la conversion de caractères est la page de codes de l'environnement local en cours.

Pour les applications client, MQCCSI\_Q\_MGR est renseigné en fonction de l'environnement local du client et non de celui du gestionnaire de files d'attente. L'exception à cette règle est lorsque vous placez un message dans une file d'attente de pont IMS ; ce qui est renvoyé, dans la zone *CodedCharSetId* de MQMD, est le CCSID du gestionnaire de files d'attente.

Vous ne devez pas utiliser la valeur spéciale suivante:

#### **MQCCSI\_APPL**

Il en résulte une valeur incorrecte dans la zone CodedCharSetId du MQMD et un code retour de MQRC\_SOURCE\_CCSD\_ERROR (ou MQRC\_FORMAT\_ERROR pour z/OS ) lorsque le message est reçu à l'aide de l'appel MQGET avec l'option MQGMO\_CONVERT.

Vous pouvez utiliser les valeurs spéciales suivantes:

#### **MQCCSI\_Q\_DIR**

Les données de type caractère du message se trouvent dans le jeu de caractères du gestionnaire de files d'attente.

Dans les appels MQPUT et MQPUT1 , le gestionnaire de files d'attente modifie cette valeur dans le MQMD qui est envoyé avec le message avec l'identificateur de jeu de caractères true du gestionnaire de files d'attente. Par conséquent, la valeur MQCCSI\_Q\_MGR n'est jamais renvoyée par l'appel MQGET.

#### **MQCCSI\_DEFAULT**

Le CodedCharSetId des données de la zone *String* est défini par la zone CodedCharSetId dans la structure d'en-tête qui précède la structure MQCFH, ou par la zone CodedCharSetId dans le MQMD si le MQCFH est au début du message.

#### **MQCCSI\_HÉRITER**

Les données de type caractère du message se trouvent dans le même jeu de caractères que cette structure ; il s'agit du jeu de caractères du gestionnaire de files d'attente. (Pour MQMD uniquement, MQCCSI\_INHERIT a la même signification que MQCCSI\_Q\_MGR).

Le gestionnaire de files d'attente remplace cette valeur dans le MQMD envoyé avec le message par l'identificateur de jeu de caractères réel de MQMD. Si aucune erreur ne se produit, la valeur MQCCSI\_INHERIT n'est pas renvoyée par l'appel MQGET.

N'utilisez pas MQCCSI\_INHERIT si la valeur de la zone PutApp1Type dans MQMD est MQAT\_BROKER.

#### **MQCCSI\_IMBRIQUÉ**

Les données de type caractère du message se trouvent dans un jeu de caractères avec l'identificateur contenu dans les données du message lui-même. Il peut y avoir n'importe quel nombre d'identificateurs de jeu de caractères imbriqués dans les données de message, s'appliquant à différentes parties des données. Cette valeur doit être utilisée pour les messages PCF (au format MQFMT\_ADMIN, MQFMT\_EVENT ou MQFMT\_PCF) qui contiennent des données dans un mélange de



jeux de caractères. Chaque structure MQCFST, MQCFSL et MQCFSF contenue dans le message PCF doit avoir un identificateur de jeu de caractères explicite spécifié et non MQCCSI\_DEFAULT.

Si un message au format MQFMT\_EMBEDDED\_PCF doit contenir des données dans une combinaison de jeux de caractères, n'utilisez pas MQCCSI\_EMBEDDED. A la place, définissez MQEPH\_CCSID\_EMBEDDED dans la zone Flags de la structure MQEPH. Cela revient à définir MQCCSI\_EMBEDDED dans la structure précédente. Chaque structure MQCFST, MQCFSL et MQCFSF contenue dans le message PCF doit alors avoir un identificateur de jeu de caractères explicite spécifié et non MQCCSI\_DEFAULT. Pour plus d'informations sur la structure MQEPH, voir «MQEPH-en-tête PCF imbriqué», à la page 377.

Indiquez cette valeur uniquement sur les appels MQPUT et MQPUT1 . S'il est spécifié dans l'appel MQGET, il empêche la conversion du message.

Dans les appels MQPUT et MQPUT1 , le gestionnaire de files d'attente modifie les valeurs MQCCSI\_Q\_MGR et MQCCSI\_INHERIT dans le MQMD envoyé avec le message comme décrit ci-dessus, mais ne modifie pas le MQMD spécifié dans l'appel MQPUT ou MQPUT1 . Aucune autre vérification n'est effectuée sur la valeur spécifiée.

Les applications qui extraient des messages doivent comparer cette zone à la valeur attendue par l'application ; si les valeurs diffèrent, l'application peut avoir besoin de convertir des données de type caractères dans le message.

Sous z/OS, la zone Encoding du MQMD est utilisée pour spécifier le codage d'entier des données de type caractères dans le corps du message, lorsque la zone CodedCharSetId du MQMD indique que la représentation du jeu de caractères dépend du codage utilisé pour les entiers binaires. Sur Multiplateformes, l'ordre des octets des données de type caractères est supposé être identique au codage d'entier natif pour la plateforme sur laquelle s'exécute le gestionnaire de files d'attente. Cela n'affecte que certains jeux de caractères multi-octets (par exemple, les jeux de caractères UTF-16 ).

Si vous spécifiez l'option MQGMO\_CONVERT dans l'appel MQGET, cette zone est une zone d'entrée-sortie. La valeur indiquée par l'application est l'identificateur de jeu de caractères codés vers lequel convertir les données de message si nécessaire. Si la conversion aboutit ou n'est pas nécessaire, la valeur reste inchangée (sauf que la valeur MQCCSI\_Q\_MGR ou MQCCSI\_INHERIT est convertie en valeur réelle). Si la conversion échoue, la valeur après l'appel MQGET représente l'identificateur de jeu de caractères codés du message non converti renvoyé à l'application.

Sinon, il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La valeur initiale de cette zone est MQCCSI\_Q\_MGR.

### **Format (MQCHAR8) pour MQMD**

Il s'agit d'un nom que l'expéditeur du message utilise pour indiquer au destinataire la nature des données du message. Tous les caractères du jeu de caractères du gestionnaire de files d'attente peuvent être spécifiés pour le nom, mais vous devez limiter le nom à ce qui suit:

- Majuscules de A à Z
- Chiffres de 0 à 9

Si d'autres caractères sont utilisés, il se peut qu'il ne soit pas possible de convertir le nom entre les jeux de caractères des gestionnaires de files d'attente d'envoi et de réception.

Mettez le nom à blanc en fonction de la longueur de la zone ou utilisez un caractère indéfini pour terminer le nom avant la fin de la zone. La valeur indéfinie et les caractères suivants sont traités comme des blancs. N'indiquez pas de nom avec des espaces de début ou des blancs imbriqués. Pour l'appel MQGET, le gestionnaire de files d'attente renvoie le nom complété par des blancs à la longueur de la zone.

Le gestionnaire de files d'attente ne vérifie pas que le nom est conforme aux recommandations décrites ci-dessus.

Les noms commençant par MQ en majuscules, en minuscules et en casse mixte ont des significations qui sont définies par le gestionnaire de files d'attente ; n'utilisez pas de noms commençant par ces lettres pour vos propres formats. Les formats intégrés du gestionnaire de files d'attente sont les suivants:

## MQFMT\_AUCUN

La nature des données n'est pas définie: les données ne peuvent pas être converties lorsque le message est extrait d'une file d'attente à l'aide de l'option MQGMO\_CONVERT.

Si vous spécifiez MQGMO\_CONVERT dans l'appel MQGET et que le jeu de caractères ou le codage des données du message diffère de celui spécifié dans le paramètre **MsgDesc**, le message est renvoyé avec les codes achèvement et anomalie suivants (en supposant qu'il n'y ait pas d'autres erreurs):

- Code achèvement MQCC\_WARNING et code anomalie MQRC\_FORMAT\_ERROR si les données MQFMT\_NONE se trouve au début du message.
- Code achèvement MQCC\_OK et code anomalie MQRC\_NONE si les données MQFMT\_NONE sont à la fin du message (c'est-à-dire précédées d'une ou de plusieurs structures d'en-tête MQ). Les structures d'en-tête MQ sont converties au jeu de caractères et au codage demandés dans ce cas.

Pour le langage de programmation C, la constante MQFMT\_NONE\_ARRAY est également définie ; elle a la même valeur que MQFMT\_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.


## MQFMT\_ADMIN

Le message est un message de demande ou de réponse du serveur de commandes au format PCF (Programmable Command Format). Les messages de ce format peuvent être convertis si l'option MQGMO\_CONVERT est spécifiée dans l'appel MQGET. Pour plus d'informations sur l'utilisation des messages au format de commande programmable, voir [Utilisation des formats de commande programmables](#).

Pour le langage de programmation C, la constante MQFMT\_ADMIN\_ARRAY est également définie ; elle a la même valeur que MQFMT\_ADMIN, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

## MQFMT\_CICS

Les données de message commencent par l'en-tête d'informations CICS MQCIH, suivi des données d'application. Le nom de format des données d'application est donné par la zone Format dans la structure MQCIH.

 Sous z/OS, spécifiez l'option MQGMO\_CONVERT dans l'appel MQGET pour convertir les messages au format MQFMT\_CICS.

Pour le langage de programmation C, la constante MQFMT\_CICS\_ARRAY est également définie ; elle a la même valeur que MQFMT\_CICS, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

## MQFMT\_COMMAND\_1

Le message est un message de réponse du serveur de commandes MQSC contenant le nombre d'objets, le code achèvement et le code anomalie. Les messages de ce format peuvent être convertis si l'option MQGMO\_CONVERT est spécifiée dans l'appel MQGET.

Pour le langage de programmation C, la constante MQFMT\_COMMAND\_1\_ARRAY est également définie ; elle a la même valeur que MQFMT\_COMMAND\_1, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

## MQFMT\_COMMAND\_2

Le message est un message de réponse du serveur de commandes MQSC contenant des informations sur les objets demandés. Les messages de ce format peuvent être convertis si l'option MQGMO\_CONVERT est spécifiée dans l'appel MQGET.

Pour le langage de programmation C, la constante MQFMT\_COMMAND\_2\_ARRAY est également définie ; elle a la même valeur que MQFMT\_COMMAND\_2, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

## MQFMT\_DEAD\_LETTER\_HEADER

Les données de message commencent par l'en-tête de rebut MQDLH. Les données du message d'origine suivent immédiatement la structure MQDLH. Le nom de format des données de message d'origine est donné par la zone Format dans la structure MQDLH ; pour plus de détails sur cette structure, voir «MQDLH-En-tête de rebut», à la page 364. Les messages de ce format peuvent être convertis si l'option MQGMO\_CONVERT est spécifiée dans l'appel MQGET.

Les rapports COA et COD ne sont pas générés pour les messages dont le Format est MQFMT\_DEAD\_LETTER\_HEADER.

Pour le langage de programmation C, la constante MQFMT\_DEAD\_LETTER\_HEADER\_ARRAY est également définie ; elle a la même valeur que MQFMT\_DEAD\_LETTER\_HEADER, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

### MQFMT\_DIST\_HEADER

Les données de message commencent par l'en-tête de liste de distribution MQDH ; cela inclut les tableaux des enregistrements MQOR et MQPMR. L'en-tête de liste de distribution peut être suivi de données supplémentaires. Le format des données supplémentaires (le cas échéant) est fourni par la zone *Format* dans la structure MQDH ; voir «En-tête MQDH-Distribution», à la page 358 pour plus de détails sur cette structure. Les messages au format MQFMT\_DIST\_HEADER peuvent être convertis si l'option MQGMO\_CONVERT est spécifiée dans l'appel MQGET.

Ce format est pris en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour IBM MQ MQI clients connecté à ces systèmes.

Pour le langage de programmation C, la constante MQFMT\_DIST\_HEADER\_ARRAY est également définie ; elle a la même valeur que MQFMT\_DIST\_HEADER, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### MQFMT\_EMBEDDED\_PCF

Format d'un message de suivi, à condition que la valeur de la commande PCF soit définie sur MQCMD\_TRACE\_ROUTE. L'utilisation de ce format permet d'envoyer des données utilisateur avec le message de trace, à condition que leurs applications puissent s'adapter aux paramètres PCF précédents.

L'en-tête PCF doit être le premier en-tête, sinon le message ne sera pas traité comme un message de suivi. Cela signifie que le message ne peut pas faire partie d'un groupe et que les messages de trace-route ne peuvent pas être segmentés. Si un message de trace-route est envoyé dans un groupe, le message est rejeté avec le code anomalie MQRC\_MSG\_NOT\_ALLOWED\_IN\_GROUP.

Notez que MQFMT\_ADMIN peut également être utilisé pour le format d'un message de routage de trace, mais dans ce cas, aucune donnée utilisateur ne peut être envoyée avec le message de routage de trace.

### MQFMT\_EVENT

Le message est un message d'événement MQ qui signale un événement qui s'est produit. Les messages d'événement ont la même structure que les commandes programmables ; voir [Messages de commande PCF](#) pour plus d'informations sur cette structure et [Surveillance des événements](#) pour plus d'informations sur les événements.

Les messages d'événement Version-1 peuvent être convertis dans tous les environnements si l'option MQGMO\_CONVERT est spécifiée dans l'appel MQGET. Les messages d'événement Version-2 peuvent être convertis uniquement sur z/OS.

Pour le langage de programmation C, la constante MQFMT\_EVENT\_ARRAY est également définie ; elle a la même valeur que MQFMT\_EVENT, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### MQFMT\_IMS

Les données de message commencent par l'en-tête d'information IMS MQIIH, qui est suivi des données d'application. Le nom de format des données d'application est donné par la zone *Format* dans la structure MQIIH.

Pour plus de détails sur la façon dont la structure MQIIH est gérée lors de l'utilisation de MQGET avec MQGMO\_CONVERT, voir «Format (MQCHAR8) pour MQIIH», à la page 427 et «Format ReplyTo(MQCHAR8) pour MQIIH», à la page 428.

Pour le langage de programmation C, la constante MQFMT\_IMS\_ARRAY est également définie ; elle a la même valeur que MQFMT\_IMS, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **MQFMT\_IMS\_VAR\_STRING**

Le message est une chaîne de variable IMS , qui est une chaîne au format 11zzccc, où:

#### **11**

est une zone de longueur de 2 octets spécifiant la longueur totale de l'élément de chaîne de variable IMS . Cette longueur est égale à la longueur de 11 (2 octets), plus la longueur de zz (2 octets), plus la longueur de la chaîne de caractères elle-même. 11 est un entier binaire à 2 octets dans le codage spécifié par la zone Encoding .

#### **zz**

est une zone à 2 octets contenant des indicateurs significatifs pour IMS. zz est une chaîne d'octets composée de deux zones MQBYTE et est transmise sans modification de l'expéditeur au destinataire (c'est-à-dire que zz n'est soumis à aucune conversion).

#### **ccc**

est une chaîne de caractères de longueur variable contenant 11-4 caractères. ccc est dans le jeu de caractères spécifié par la zone CodedCharSetId .

Sous z/OS, les données de message peuvent être constituées d'une séquence de chaînes de variables IMS butted ensemble, chaque chaîne étant de la forme 11zzccc. Aucun octet ne doit être ignoré entre les chaînes de variable IMS successives. Cela signifie que si la première chaîne a une longueur impaire, la deuxième chaîne sera mal alignée, c'est-à-dire qu'elle ne commencera pas sur une limite qui est un multiple de deux. Faites attention lorsque vous construisez de telles chaînes sur des machines qui nécessitent l'alignement de types de données élémentaires.

Utilisez l'option MQGMO\_CONVERT dans l'appel MQGET pour convertir les messages dont le format est MQFMT\_IMS\_VAR\_STRING.

Pour le langage de programmation C, la constante MQFMT\_IMS\_VAR\_STRING\_ARRAY est également définie ; elle a la même valeur que MQFMT\_IMS\_VAR\_STRING, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **MQFMT\_MD\_EXTENSION**

Les données de message commencent par l'extension de descripteur de message MQMDE et sont éventuellement suivies par d'autres données (généralement les données de message d'application). Le nom de format, le jeu de caractères et le codage des données qui suivent MQMDE sont fournis par les zones Format, CodedCharSetId et Encoding dans MQMDE. Pour plus de détails sur cette structure, voir «MQMDE-Extension de descripteur de message», à la page 492 . Les messages de ce format peuvent être convertis si l'option MQGMO\_CONVERT est spécifiée dans l'appel MQGET.

Pour le langage de programmation C, la constante MQFMT\_MD\_EXTENSION\_ARRAY est également définie ; elle a la même valeur que MQFMT\_MD\_EXTENSION, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

### **MQFMT\_PCF**

Le message est un message défini par l'utilisateur qui est conforme à la structure d'un message PCF (Programmable Command Format). Les messages de ce format peuvent être convertis si l'option MQGMO\_CONVERT est spécifiée dans l'appel MQGET. Pour plus d'informations sur l'utilisation des messages au format de commande programmable, voir [Utilisation des formats de commande programmables](#) .

Pour le langage de programmation C, la constante MQFMT\_PCF\_ARRAY est également définie ; elle a la même valeur que MQFMT\_PCF, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **MQFMT\_REF\_MSG\_HEADER**

Les données de message commencent par l'en-tête de message de référence MQRMH et sont éventuellement suivies d'autres données. Le nom de format, le jeu de caractères et le codage des

données sont donnés par les zones `Format`, `CodedCharSetIdet Encoding` dans `MQRMH`. Pour plus de détails sur cette structure, voir «[MQRMH-En-tête de message de référence](#)», à la page 574 . Les messages de ce format peuvent être convertis si l'option `MQGMO_CONVERT` est spécifiée dans l'appel `MQGET`.

Ce format est pris en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour IBM MQ MQI clients connecté à ces systèmes.

Pour le langage de programmation C, la constante `MQFMT_REF_MSG_HEADER_ARRAY` est également définie ; elle a la même valeur que `MQFMT_REF_MSG_HEADER`, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **MQFMT\_RF\_HEADER**

Les données de message commencent par les règles et l'en-tête de formatage `MQRFH`, et sont éventuellement suivies par d'autres données. Le nom de format, le jeu de caractères et le codage des données (le cas échéant) sont fournis par les zones `Format`, `CodedCharSetIdet Encoding` dans le `MQRFH`. Les messages de ce format peuvent être convertis si l'option `MQGMO_CONVERT` est spécifiée dans l'appel `MQGET`.

Pour le langage de programmation C, la constante `MQFMT_RF_HEADER_ARRAY` est également définie ; elle a la même valeur que `MQFMT_RF_HEADER`, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **MQFMT\_RF\_HEADER\_2**

Les données de message commencent par les règles version-2 et l'en-tête de formatage `MQRFH2`, et sont éventuellement suivies par d'autres données. Le nom de format, le jeu de caractères et le codage des données facultatives (le cas échéant) sont fournis par les zones `Format`, `CodedCharSetIdet Encoding` dans `MQRFH2`. Les messages de ce format peuvent être convertis si l'option `MQGMO_CONVERT` est spécifiée dans l'appel `MQGET`.

Pour le langage de programmation C, la constante `MQFMT_RF_HEADER_2_ARRAY` est également définie ; elle a la même valeur que `MQFMT_RF_HEADER_2`, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

### **MQFMT\_STRING**

Les données de message d'application peuvent être soit une chaîne SBCS (jeu de caractères codé sur un octet), soit une chaîne DBCS (jeu de caractères codé sur deux octets). Les messages de ce format peuvent être convertis si l'option `MQGMO_CONVERT` est spécifiée dans l'appel `MQGET`.

Pour le langage de programmation C, la constante `MQFMT_STRING_ARRAY` est également définie ; elle a la même valeur que `MQFMT_STRING`, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.


### **MQFMT\_TRIGGER**

Le message est un message de déclenchement, décrit par la structure `MQTM`. Pour plus de détails sur cette structure, voir «[MQTM-Message de déclenchement](#)», à la page 627 . Les messages de ce format peuvent être convertis si l'option `MQGMO_CONVERT` est spécifiée dans l'appel `MQGET`.

Pour le langage de programmation C, la constante `MQFMT_TRIGGER_ARRAY` est également définie ; elle a la même valeur que `MQFMT_TRIGGER`, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

## MQFMT\_WORK\_INFO\_HEADER

Les données de message commencent par l'en-tête d'informations de travail MQWIH, qui est suivi des données d'application. Le nom de format des données d'application est donné par la zone Format dans la structure MQWIH.

 Sous z/OS, spécifiez l'option MQGMO\_CONVERT dans l'appel MQGET pour convertir les données utilisateur dans les messages au format MQFMT\_WORK\_INFO\_HEADER. Toutefois, la structure MQWIH elle-même est toujours renvoyée dans le jeu de caractères et le codage du gestionnaire de files d'attente (c'est-à-dire que la structure MQWIH est convertie, que l'option MQGMO\_CONVERT soit spécifiée ou non).

Pour le langage de programmation C, la constante MQFMT\_WORK\_INFO\_HEADER\_ARRAY est également définie ; elle a la même valeur que MQFMT\_WORK\_INFO\_HEADER, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

## MQFMT\_XMIT\_Q\_HEADER

Les données de message commencent par l'en-tête de file d'attente de transmission MQXQH. Les données du message d'origine suivent immédiatement la structure MQXQH. Le nom de format des données de message d'origine est donné par la zone Format de la structure MQMD, qui fait partie de l'en-tête de file d'attente de transmission MQXQH. Pour plus de détails sur cette structure, voir «MQXQH-en-tête de file d'attente de transmission», à la page 647 .

Les rapports COA et COD ne sont pas générés pour les messages dont le Format est MQFMT\_XMIT\_Q\_HEADER.

Pour le langage de programmation C, la constante MQFMT\_XMIT\_Q\_HEADER\_ARRAY est également définie ; elle a la même valeur que MQFMT\_XMIT\_Q\_HEADER, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La longueur de cette zone est indiquée par MQ\_FORMAT\_LENGTH. La valeur initiale de cette zone est MQFMT\_NONE.

## Priorité (MQLONG) pour MQMD

Pour les appels MQPUT et MQPUT1 , la valeur doit être supérieure ou égale à zéro ; zéro est la priorité la plus basse. La valeur spéciale suivante peut également être utilisée:

### MQPRI\_PRIORITY\_AS\_Q\_DEF

- Si la file d'attente est une file d'attente de cluster, la priorité du message est extraite de l'attribut **DefPriority** défini sur le gestionnaire de files d'attente *destination* qui possède l'instance particulière de la file d'attente dans laquelle le message est placé.

Lorsqu'il existe plusieurs instances de la file d'attente de cluster et que cet attribut est différent pour chaque instance, la valeur définie dans l'une des instances est utilisée, mais il n'est pas possible de prévoir l'instance qui sera choisie. Par conséquent, associez cet attribut à la même valeur dans toutes les instances. Sinon, le message d'erreur AMQ9407 est émis dans les journaux du gestionnaire de files d'attente. Voir aussi [How are destination object attributes resolved for aliases, remote and cluster queues?](#)

La valeur de *DefPriority* est copiée dans la zone *Priority* lorsque le message est placé dans la file d'attente de destination. Si *DefPriority* est modifié ultérieurement, les messages qui ont déjà été placés dans la file d'attente ne sont pas affectés.

- Si la file d'attente n'est pas une file d'attente de cluster, la priorité du message est extraite de l'attribut **DefPriority** défini sur le gestionnaire de files d'attente *local* , même si le gestionnaire de files d'attente de destination est distant.

S'il existe plusieurs définitions dans le chemin de résolution de nom de file d'attente, la priorité par défaut est extraite de la valeur de cet attribut dans la *première* définition du chemin. Ce nom peut être :

- Une file d'attente alias

- Une file d'attente locale
- Définition locale d'une file d'attente éloignée
- Un alias de gestionnaire de files d'attente
- Une file d'attente de transmission (par exemple, la file d'attente *DefXmitQName* )

La valeur de *DefPriority* est copiée dans la zone *Priority* lorsque le message est inséré. Si *DefPriority* est modifié par la suite, les messages qui ont déjà été insérés ne sont pas affectés.

La valeur renvoyée par l'appel MQGET est toujours supérieure ou égale à zéro ; la valeur MQPRI\_PRIORITY\_AS\_Q\_DEF n'est jamais renvoyée.

Si un message est inséré avec une priorité supérieure à la valeur maximale prise en charge par le gestionnaire de files d'attente local (cette valeur maximale est donnée par l'attribut de gestionnaire de files d'attente **MaxPriority** ), le message est accepté par le gestionnaire de files d'attente, mais placé dans la file d'attente avec la priorité maximale du gestionnaire de files d'attente ; l'appel MQPUT ou MQPUT1 se termine avec MQCC\_WARNING et le code anomalie MQRC\_PRIORITY\_DÉPASSS\_MAXIMUM. Toutefois, la zone *Priority* conserve la valeur spécifiée par l'application qui a inséré le message.

Sous z/OS, si un message dont le MsgSeqnuméro 1 est inséré dans une file d'attente dont la séquence de distribution des messages est MQMDS\_PRIORITY et dont le type d'index est MQIT\_GROUP\_ID, la file d'attente peut traiter le message avec une priorité différente. Si le message a été placé dans la file d'attente avec une priorité de 0 ou 1, il est traité comme s'il avait une priorité de 2. En effet, l'ordre des messages placés dans ce type de file d'attente est optimisé pour permettre des tests de complétude de groupe efficaces. Pour plus d'informations sur la séquence de distribution des messages MQMDS\_PRIORITY et le type d'index MQIT\_GROUP\_ID, voir Attribut de séquenceMsgDelivery.

Lors de la réponse à un message, les applications doivent utiliser la priorité du message de demande pour le message de réponse. Dans d'autres cas, la spécification de MQPRI\_PRIORITY\_AS\_Q\_DEF permet d'optimiser les priorités sans modifier l'application.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La valeur initiale de cette zone est MQPRI\_PRIORITY\_AS\_Q\_DEF.

### **Persistance (MQLONG) pour MQMD**

Indique si le message survit aux échecs du système et aux redémarrages du gestionnaire de files d'attente. Pour les appels MQPUT et MQPUT1 , la valeur doit être l'une des suivantes:

#### **MQPER\_PERSISTANT**

Le message survit aux échecs du système et aux redémarrages du gestionnaire de files d'attente. Une fois que le message a été inséré et que l'unité de travail dans laquelle il a été inséré a été validée (si le message est inséré dans une unité de travail), le message est conservé dans la mémoire secondaire. Il y reste jusqu'à ce que le message soit supprimé de la file d'attente et que l'unité de travail dans laquelle il a été obtenu ait été validée (si le message est extrait dans le cadre d'une unité de travail).

Lorsqu'un message persistant est envoyé à une file d'attente éloignée, un mécanisme de stockage et de transfert conserve le message sur chaque gestionnaire de files d'attente le long de la route menant à la destination, jusqu'à ce que le message soit connu comme étant arrivé sur le gestionnaire de files d'attente suivant.

Les messages persistants ne peuvent pas être placés sur:

- Files d'attente dynamiques temporaires
- Files d'attente partagées qui sont mappées à un objet CFSTRUCT au niveau CFLEVEL (2) ou à un niveau inférieur, ou où l'objet CFSTRUCT est défini en tant que RECOVER (NO).

Les messages persistants peuvent être placés dans des files d'attente dynamiques permanentes et dans des files d'attente prédéfinies.

#### **MQPER\_NON\_PERSISTENT**

Le message ne survit généralement pas aux défaillances du système ou aux redémarrages du gestionnaire de files d'attente. Cela s'applique même si une copie intacte du message est détectée sur la mémoire secondaire lors du redémarrage du gestionnaire de files d'attente.

Dans le cas des files d'attente NPMCLASS (HIGH), les messages non persistants survivent à un arrêt et à un redémarrage normal du gestionnaire de files d'attente.

Dans le cas de files d'attente partagées, les messages non persistants survivent aux redémarrages du gestionnaire de files d'attente dans le groupe de partage de files d'attente, mais ne survivent pas aux échecs de l'unité de couplage utilisée pour stocker les messages dans les files d'attente partagées.

## **MQPER\_PERSISTENCE\_AS\_Q\_DEF**

- Si la file d'attente est une file d'attente de cluster, la persistance du message est extraite de l'attribut **DefPersistence** défini sur le gestionnaire de files d'attente *destination* qui possède l'instance particulière de la file d'attente dans laquelle le message est placé.

Lorsqu'il existe plusieurs instances de la file d'attente de cluster et que cet attribut est différent pour chaque instance, la valeur définie dans l'une des instances est utilisée, mais il n'est pas possible de prévoir l'instance qui sera choisie. Par conséquent, associez cet attribut à la même valeur dans toutes les instances. Sinon, le message d'erreur AMQ9407 est émis dans les journaux du gestionnaire de files d'attente. Voir aussi [How are destination object attributes resolved for aliases, remote and cluster queues?](#)

La valeur de *DefPersistence* est copiée dans la zone *Persistence* lorsque le message est placé dans la file d'attente de destination. Si *DefPersistence* est modifié ultérieurement, les messages qui ont déjà été placés dans la file d'attente ne sont pas affectés.

- Si la file d'attente n'est pas une file d'attente de cluster, la persistance du message est extraite de l'attribut **DefPersistence** défini sur le gestionnaire de files d'attente *local*, même si le gestionnaire de files d'attente de destination est distant.

S'il existe plusieurs définitions dans le chemin de résolution de nom de file d'attente, la persistance par défaut est extraite de la valeur de cet attribut dans la *première* définition du chemin. Ce nom peut être :

- Une file d'attente alias
- Une file d'attente locale
- Définition locale d'une file d'attente éloignée
- Un alias de gestionnaire de files d'attente
- Une file d'attente de transmission (par exemple, la file d'attente *DefXmitQName*)

La valeur de *DefPersistence* est copiée dans la zone *Persistence* lorsque le message est inséré. Si *DefPersistence* est modifié par la suite, les messages qui ont déjà été insérés ne sont pas affectés.

Les messages persistants et non persistants peuvent exister dans la même file d'attente.

Lors de la réponse à un message, les applications doivent utiliser la persistance du message de demande pour le message de réponse.

Pour un appel MQGET, la valeur renvoyée est MQPER\_PERSISTENT ou MQPER\_NOT\_PERSISTENT.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1. La valeur initiale de cette zone est MQPER\_PERSISTENCE\_AS\_Q\_DEF.

## **MsgId (MQBYTE24) pour MQMD**

Il s'agit d'une chaîne d'octets utilisée pour distinguer un message d'un autre. En règle générale, deux messages ne doivent pas avoir le même identificateur de message, bien que cela ne soit pas interdit par le gestionnaire de files d'attente. L'identificateur de message est une propriété permanente du message et est conservé lors des redémarrages du gestionnaire de files d'attente. Etant donné que l'identificateur de message est une chaîne d'octets et non une chaîne de caractères, l'identificateur de message n'est pas converti entre les jeux de caractères lorsque le message passe d'un gestionnaire de files d'attente à un autre.

Pour les appels MQPUT et MQPUT1, si MQMI\_NONE ou MQPMO\_NEW\_MSG\_ID est spécifié par l'application, le gestionnaire de files d'attente génère un identificateur de message unique<sup>3</sup> lorsque le



message est inséré et le place dans le descripteur de message envoyé avec le message. Le gestionnaire de files d'attente renvoie également cet identificateur de message dans le descripteur de message appartenant à l'application émettrice. L'application peut utiliser cette valeur pour enregistrer des informations sur des messages particuliers et pour répondre à des requêtes provenant d'autres parties de l'application.

Si le message est inséré dans une rubrique, le gestionnaire de files d'attente génère des identificateurs de message uniques pour chaque message publié. Si MQPMO\_NEW\_MSG\_ID est spécifié par l'application, le gestionnaire de files d'attente génère un identificateur de message unique à renvoyer en sortie. Si MQMI\_NONE est spécifié par l'application, la valeur de la zone *MsgId* dans le MQMD reste inchangée lors du retour de l'appel.

Voir la description de MQPMO\_RETAIN dans «Options (MQLONG) pour MQPMO», à la page 527 pour plus de détails sur les publications conservées.

Si le message est inséré dans une liste de distribution, le gestionnaire de files d'attente génère des identificateurs de message uniques si nécessaire, mais la valeur de la zone *MsgId* dans MQMD est inchangée lors du retour de l'appel, même si MQMI\_NONE ou MQPMO\_NEW\_MSG\_ID a été spécifié. Si l'application a besoin de connaître les identificateurs de message générés par le gestionnaire de files d'attente, elle doit fournir des enregistrements MQPMR contenant la zone *MsgId*.

L'application émettrice peut également spécifier une valeur pour l'identificateur de message autre que MQMI\_NON; cela empêche le gestionnaire de files d'attente de générer un identificateur de message unique. Une application qui transfère un message peut l'utiliser pour propager l'identificateur du message d'origine.

Le gestionnaire de files d'attente n'utilise pas cette zone sauf pour:

- Générer une valeur unique si demandée, comme décrit ci-dessus
- Distribuer la valeur à l'application qui émet la demande d'obtention du message
- Copier la valeur dans la zone *CorrelId* de tout message de rapport généré à propos de ce message (en fonction des options *Report*)

Lorsque le gestionnaire de files d'attente ou un agent MCA génère un message de rapport, il définit la zone *MsgId* de la manière spécifiée par la zone *Report* du message d'origine, à savoir MQRO\_NEW\_MSG\_ID ou MQRO\_PASS\_MSG\_ID. Les applications qui génèrent des messages de rapport doivent également effectuer cette opération.

Pour l'appel MQGET, *MsgId* est l'une des cinq zones pouvant être utilisées pour extraire un message particulier de la file d'attente. Normalement, l'appel MQGET renvoie le message suivant dans la file d'attente, mais un message particulier peut être obtenu en spécifiant un ou plusieurs des cinq critères de sélection, dans n'importe quelle combinaison ; ces zones sont les suivantes:

- *MsgId*
- *CorrelId*
- *GroupId*
- *MsgSeqNumber*
- *Offset*

---

<sup>3</sup> Un *MsgId* généré par le gestionnaire de files d'attente se compose d'un identificateur de produit de 4 octets (AMQ- ou CSQ- en ASCII ou EBCDIC, où - représente un caractère blanc), suivi d'une implémentation spécifique au produit d'une chaîne unique. Dans IBM MQ, il contient les 12 premiers caractères du nom du gestionnaire de files d'attente et une valeur dérivée de l'horloge système. Tous les gestionnaires de files d'attente qui peuvent intercommuniquer doivent donc avoir des noms qui diffèrent dans les 12 premiers caractères, afin de s'assurer que les identificateurs de message sont uniques. La possibilité de générer une chaîne unique dépend également du fait que l'horloge système n'est pas modifiée en amont. Pour éviter qu'un identificateur de message généré par le gestionnaire de files d'attente ne soit dupliqué par l'application, celle-ci doit éviter de générer des identificateurs dont les caractères initiaux sont compris entre A et I en ASCII ou EBCDIC (X'41' à X'49' et X'C1' à X'C9'). Toutefois, l'application n'est pas empêchée de générer des identificateurs avec des caractères initiaux dans ces plages.

L'application définit une ou plusieurs de ces zones sur les valeurs requises, puis définit les options de correspondance MQMO\_\* correspondantes dans la zone *MatchOptions* de MQGMO pour utiliser ces zones comme critères de sélection. Seuls les messages dont les valeurs sont indiquées dans ces zones peuvent être extraits. La valeur par défaut de la zone *MatchOptions* (si elle n'est pas modifiée par l'application) correspond à la fois à l'identificateur de message et à l'identificateur de corrélation.

Sous z/OS, les critères de sélection que vous pouvez utiliser sont limités par le type d'index utilisé pour la file d'attente. Pour plus de détails, voir l'attribut de file d'attente **IndexType**.

Normalement, le message renvoyé est le *premier* message de la file d'attente qui répond aux critères de sélection. Mais si MQGMO\_BROWSE\_NEXT est spécifié, le message renvoyé est le message *suivant* qui répond aux critères de sélection ; l'analyse de ce message commence par le message *suivant* la position actuelle du curseur.

**Remarque :** La file d'attente est analysée séquentiellement à la recherche d'un message qui répond aux critères de sélection. Par conséquent, les temps d'extraction sont plus lents que si aucun critère de sélection n'est spécifié, en particulier si de nombreux messages doivent être analysés avant qu'un message approprié ne soit trouvé. Les exceptions sont les suivantes:

- **Multi** un appel MQGET par *CorrelId* sur des multiplateformes 64 bits où l'index *CorrelId* élimine la nécessité d'effectuer une analyse séquentielle réelle.
- **z/OS** un appel MQGET par *IndexType* sous z/OS.

Dans ces deux cas, les performances d'extraction sont améliorées.

Pour plus d'informations sur l'utilisation des critères de sélection dans diverses situations, voir [Tableau 495](#), à la page 409.

La spécification de MQMI\_NONE comme identificateur de message a le même effet que la non-spécification de MQMO\_MATCH\_MSG\_ID, c'est-à-dire *tout* identificateur de message correspondant.

Cette zone est ignorée si l'option MQGMO\_MSG\_UNDER\_CURSOR est spécifiée dans le paramètre **GetMsgOpts** de l'appel MQGET.

En cas de retour d'un appel MQGET, la zone *MsgId* est définie sur l'identificateur du message renvoyé (le cas échéant).

La valeur spéciale suivante peut être utilisée:

#### **MQMI\_AUCUN**

Aucun identificateur de message n'est indiqué.

La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQMI\_NONE\_ARRAY est également définie ; elle a la même valeur que MQMI\_NONE, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

Il s'agit d'une zone d'entrée-sortie pour les appels MQGET, MQPUT et MQPUT1. La longueur de cette zone est indiquée par MQ\_MSG\_ID\_LENGTH. La valeur initiale de cette zone est MQMI\_NONE.

#### **CorrelId (MQBYTE24) pour MQMD**

La zone *CorrelId* est une propriété de l'en-tête de message qui peut être utilisée pour identifier un message ou un groupe de messages spécifique.

Il s'agit d'une chaîne d'octets que l'application peut utiliser pour relier un message à un autre ou pour relier le message à un autre travail effectué par l'application. L'identificateur de corrélation est une propriété permanente du message et est conservé lors des redémarrages du gestionnaire de files d'attente. Etant donné que l'identificateur de corrélation est une chaîne d'octets et non une chaîne de caractères, l'identificateur de corrélation n'est pas converti entre les jeux de caractères lorsque le message est transmis d'un gestionnaire de files d'attente à un autre.

Pour les appels MQPUT et MQPUT1, l'application peut spécifier n'importe quelle valeur. Le gestionnaire de files d'attente transmet cette valeur avec le message et la distribue à l'application qui émet la demande d'obtention du message.

Si l'application spécifie MQPMO\_NEW\_CORREL\_ID, le gestionnaire de files d'attente génère un identificateur de corrélation unique qui est envoyé avec le message et renvoyé à l'application émettrice dans la sortie de l'appel MQPUT ou MQPUT1 .

Un identificateur de corrélation généré par le gestionnaire de files d'attente se compose d'un identificateur de produit de 3 octets (AMQ ou CSQ en ASCII ou EBCDIC), suivi d'un octet réservé et d'une implémentation spécifique du produit d'une chaîne unique. Dans IBM MQ , cette chaîne d'implémentation spécifique au produit contient les 12 premiers caractères du nom du gestionnaire de files d'attente et une valeur dérivée de l'horloge système. Tous les gestionnaires de files d'attente qui peuvent intercommuniquer doivent donc avoir des noms qui diffèrent dans les 12 premiers caractères pour s'assurer que les identificateurs de message sont uniques. La possibilité de générer une chaîne unique dépend également du fait que l'horloge système n'est pas modifiée en amont. Pour éviter qu'un identificateur de message généré par le gestionnaire de files d'attente ne soit dupliqué par l'application, celle-ci doit éviter de générer des identificateurs dont les caractères initiaux sont compris entre A et I en ASCII ou EBCDIC (X'41'à X'49'et X'C1'à X'C9'). Toutefois, l'application n'est pas empêchée de générer des identificateurs avec des caractères initiaux dans ces plages.

Cet identificateur de corrélation généré est conservé avec le message s'il est conservé et il est utilisé comme identificateur de corrélation lorsque le message est envoyé en tant que publication aux abonnés qui spécifient MQCI\_NONE dans la zone SubCorrelde l'ID MQSD transmis à l'appel MQSUB. Voir [Options MQPMO](#) pour plus de détails sur les publications conservées.

Lorsque le gestionnaire de files d'attente ou un agent MCA génère un message de rapport, il définit la zone *CorrelId* de la manière spécifiée par la zone *Report* du message d'origine, MQRO\_COPY\_MSG\_ID\_TO\_CORREL\_ID ou MQRO\_PASS\_CORREL\_ID. Les applications qui génèrent des messages de rapport doivent également effectuer cette opération.

Pour l'appel MQGET, *CorrelId* est l'une des cinq zones qui peuvent être utilisées pour sélectionner un message particulier à extraire de la file d'attente. Voir la description de la zone *MsgId* pour plus de détails sur la manière de spécifier des valeurs pour cette zone.

La spécification de MQCI\_NONE comme identificateur de corrélation a le même effet que la non-spécification de MQMO\_MATCH\_CORREL\_ID, c'est-à-dire que *tout* identificateur de corrélation correspond.

Si l'option MQGMO\_MSG\_UNDER\_CURSOR est spécifiée dans le paramètre **GetMsgOpts** de l'appel MQGET, cette zone est ignorée.

En cas de retour d'un appel MQGET, la zone *CorrelId* est définie sur l'identificateur de corrélation du message renvoyé (le cas échéant).

Les valeurs spéciales suivantes peuvent être utilisées:

### **MQCI\_NONE**

Aucun identificateur de corrélation n'est indiqué.

La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQCI\_NONE\_ARRAY est également définie ; elle a la même valeur que MQCI\_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **MQCI\_NOUVEAU\_SESSION**

Le message est le début d'une nouvelle session.

Cette valeur est reconnue par CICS bridge comme indiquant le début d'une nouvelle session, c'est-à-dire le début d'une nouvelle séquence de messages.

Pour le langage de programmation C, la constante MQCI\_NEW\_SESSION\_ARRAY est également définie ; elle a la même valeur que MQCI\_NEW\_SESSION, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Pour l'appel MQGET, il s'agit d'une zone d'entrée/sortie. Pour les appels MQPUT et MQPUT1 , il s'agit d'une zone d'entrée si MQPMO\_NEW\_CORREL\_ID n'est pas spécifié et d'une zone de sortie si MQPMO\_NEW\_CORREL\_ID est spécifié. La longueur de cette zone est indiquée par MQ\_CORREL\_ID\_LENGTH. La valeur initiale de cette zone est MQCI\_NONE.

### Remarque :

Vous ne pouvez pas transmettre l'identificateur de corrélation d'une publication dans une hiérarchie. Cette zone est utilisée par le gestionnaire de files d'attente.

### **BackoutCount (MQLONG) pour MQMD**

Il s'agit du nombre de fois où le message a été précédemment renvoyé par l'appel MQGET dans le cadre d'une unité de travail, puis annulé. Il aide l'application à détecter les erreurs de traitement basées sur le contenu des messages. Le nombre exclut les appels MQGET qui spécifient l'une des options MQGMO\_BROWSE\_\*.

La précision de ce nombre est affectée par l'attribut de file d'attente **HardenGetBackout** ; voir «Attributs des files d'attente», à la page 875.

Sous z/OS, la valeur 255 signifie que le message a été annulé 255 fois ou plus ; la valeur renvoyée n'est jamais supérieure à 255.

Il s'agit d'une zone de sortie pour l'appel MQGET. Il est ignoré pour les appels MQPUT et MQPUT1 . La valeur initiale de cette zone est 0.

### **ReplyToQ (MQCHAR48) pour MQMD**

Il s'agit du nom de la file d'attente de messages à laquelle l'application qui a émis la demande d'obtention du message envoie les messages MQMT\_REPLY et MQMT\_REPORT. Il s'agit du nom local d'une file d'attente définie sur le gestionnaire de files d'attente identifié par *ReplyToQMGr*. Cette file d'attente ne doit pas être une file d'attente modèle, bien que le gestionnaire de files d'attente émetteur ne la vérifie pas lorsque le message est inséré.

Pour les appels MQPUT et MQPUT1 , cette zone ne doit pas être vide si la zone *MsgType* a la valeur MQMT\_REQUEST ou si des messages de rapport sont demandés par la zone *Report* . Toutefois, la valeur spécifiée (ou remplacée) est transmise à l'application qui émet la demande d'obtention du message, quel que soit le type de message.

Si la zone *ReplyToQMGr* est vide, le gestionnaire de files d'attente local recherche le nom *ReplyToQ* dans ses propres définitions de file d'attente. S'il existe une définition locale d'une file d'attente éloignée portant ce nom, la valeur *ReplyToQ* dans le message transmis est remplacée par la valeur de l'attribut **RemoteQName** de la définition de la file d'attente éloignée et cette valeur est renvoyée dans le descripteur de message lorsque l'application de réception émet un appel MQGET pour le message. Si une définition locale d'une file d'attente éloignée n'existe pas, *ReplyToQ* n'est pas modifié.

Si le nom est indiqué, il peut contenir des blancs de fin ; le premier caractère nul et les caractères qui le suivent sont traités comme des blancs. Sinon, il n'est pas vérifié que le nom respecte les règles de dénomination des files d'attente ; cela est également vrai pour le nom transmis, si *ReplyToQ* est remplacé dans le message transmis. La seule vérification effectuée est qu'un nom a été spécifié, si les circonstances l'exigent.

Si une file d'attente de réponse n'est pas requise, définissez la zone *ReplyToQ* sur des blancs ou (dans le langage de programmation C) sur la chaîne NULL ou sur un ou plusieurs blancs suivis d'un caractère NULL ; ne laissez pas la zone non initialisée.

Pour l'appel MQGET, le gestionnaire de files d'attente renvoie toujours le nom complété par des blancs à la longueur de la zone.

Si un message nécessitant un message de rapport ne peut pas être distribué et que le message de rapport ne peut pas non plus être distribué dans la file d'attente spécifiée, le message d'origine et le message de rapport sont placés dans la file d'attente de rebut (message non distribué) (voir l'attribut **DeadLetterQName** décrit dans «Attributs du gestionnaire de files d'attente», à la page 835 ).

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La longueur de cette zone est indiquée par MQ\_Q\_NAME\_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

## **ReplyToGestionnaire de files d'attente (MQCHAR48) pour MQMD**

Il s'agit du nom du gestionnaire de files d'attente auquel le message de réponse ou de rapport doit être envoyé. *ReplyToQ* est le nom local d'une file d'attente définie sur ce gestionnaire de files d'attente.

Si la zone *ReplyToQMgr* est vide, le gestionnaire de files d'attente local recherche le nom *ReplyToQ* dans ses définitions de file d'attente. S'il existe une définition locale d'une file d'attente éloignée portant ce nom, la valeur *ReplyToQMgr* dans le message transmis est remplacée par la valeur de l'attribut **RemoteQMgrName** de la définition de la file d'attente éloignée et cette valeur est renvoyée dans le descripteur de message lorsque l'application de réception émet un appel MQGET pour le message. S'il n'existe pas de définition locale d'une file d'attente éloignée, le *ReplyToQMgr* transmis avec le message est le nom du gestionnaire de files d'attente local.

Si le nom est indiqué, il peut contenir des blancs de fin ; le premier caractère nul et les caractères qui le suivent sont traités comme des blancs. Sinon, il n'est pas vérifié que le nom respecte les règles de dénomination des gestionnaires de files d'attente ou que ce nom est connu du gestionnaire de files d'attente émetteur ; cela est également vrai pour le nom transmis, si *ReplyToQMgr* est remplacé dans le message transmis.

Si une file d'attente de réponse n'est pas requise, définissez la zone *ReplyToQMgr* sur des blancs ou (dans le langage de programmation C) sur la chaîne NULL ou sur un ou plusieurs blancs suivis d'un caractère NULL ; ne laissez pas la zone non initialisée.

Pour l'appel MQGET, le gestionnaire de files d'attente renvoie toujours le nom complété par des blancs à la longueur de la zone.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1. La longueur de cette zone est indiquée par MQ\_Q\_MGR\_NAME\_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

## **UserIdentifieur (MQCHAR12) pour MQMD**

Il fait partie du **contexte d'identité** du message. Pour plus d'informations sur le contexte de message, voir «MQMD-Descripteur de message», à la page 438 et [Contexte de message](#).

*UserIdentifieur* indique l'ID utilisateur de l'application à l'origine du message. Le gestionnaire de files d'attente traite ces informations comme des données de type caractères, mais ne définit pas leur format.

Une fois qu'un message a été reçu, utilisez *UserIdentifieur* dans la zone *AlternateUserId* du paramètre **ObjDesc** d'un appel MQOPEN ou MQPUT1 ultérieur pour effectuer la vérification d'autorisation de l'utilisateur *UserIdentifieur* au lieu de l'application effectuant l'ouverture.

Lorsque le gestionnaire de files d'attente génère ces informations pour un appel MQPUT ou MQPUT1 :

- Sous z/OS, le gestionnaire de files d'attente utilise le paramètre *AlternateUserId* du paramètre **ObjDesc** de l'appel MQOPEN ou MQPUT1 si l'option MQOO\_ALTERNATE\_USER\_AUTHORITY ou MQPMO\_ALTERNATE\_USER\_AUTHORITY a été spécifiée. Si l'option appropriée n'a pas été spécifiée, le gestionnaire de files d'attente utilise un identificateur utilisateur déterminé à partir de l'environnement.
- Dans les autres environnements, le gestionnaire de files d'attente utilise toujours un identificateur utilisateur déterminé à partir de l'environnement.

Lorsque l'ID utilisateur est déterminé à partir de l'environnement:

- Sous z/OS, le gestionnaire de files d'attente utilise:
  - Pour MVS (par lots), identificateur utilisateur de la carte de travail JES ou de la tâche démarrée
  - Pour TSO, l'ID utilisateur est propagé au travail lors de la soumission du travail
  - Pour CICS, l'identificateur utilisateur associé à la tâche
  - Pour IMS, l'identificateur d'utilisateur dépend du type d'application:
    - Pour :
      - Régions BMP sans message

- Régions IFP sans message
- Régions BMP de message et IFP de message qui n'ont pas émis d'appel GU réussi

le gestionnaire de files d'attente utilise l'ID utilisateur de la carte JOB JES de la région ou l'ID utilisateur TSO. S'ils sont vides ou nuls, ils utilisent le nom du bloc de spécification de programme (PSB).

- Pour :

- Les régions BMP de message et IFP de message qui *ont* émis un appel GU réussi
- Régions MPP

le gestionnaire de files d'attente utilise l'un des éléments suivants:

- ID utilisateur connecté associé au message
  - Nom du terminal logique (LTERM)
  - Identificateur utilisateur de la carte de travail JES de la région
  - ID utilisateur TSO
  - Nom du bloc de spécification de programme
- Sous IBM i, le gestionnaire de files d'attente utilise le nom du profil utilisateur associé au travail d'application.
  - Sous AIX and Linux, le gestionnaire de files d'attente utilise:
    - Nom de connexion de l'application
    - ID utilisateur effectif du processus si aucune connexion n'est disponible
    - Identificateur utilisateur associé à la transaction, si l'application est une transaction CICS
  - Sur les systèmes Windows , le gestionnaire de files d'attente utilise les 12 premiers caractères du nom d'utilisateur connecté.

Cette zone est normalement une zone de sortie générée par le gestionnaire de files d'attente, mais pour un appel MQPUT ou MQPUT1 , vous pouvez définir cette zone comme zone d'entrée-sortie et spécifier la zone UserIdentification au lieu de laisser le gestionnaire de files d'attente générer ces informations. Indiquez MQPMO\_SET\_IDENTITY\_CONTEXT ou MQPMO\_SET\_ALL\_CONTEXT dans le paramètre PutMsgOpts et indiquez un ID utilisateur dans la zone UserIdentifier si vous ne souhaitez pas que le gestionnaire de files d'attente génère la zone UserIdentifier pour un appel MQPUT ou MQPUT1 .

Pour les appels MQPUT et MQPUT1 , il s'agit d'une zone d'entrée-sortie si MQPMO\_SET\_IDENTITY\_CONTEXT ou MQPMO\_SET\_ALL\_CONTEXT est spécifié dans le paramètre **PutMsgOpts** . Toutes les informations qui suivent un caractère null dans la zone sont supprimées. Le gestionnaire de files d'attente convertit le caractère null et les caractères suivants en blancs. Si MQPMO\_SET\_IDENTITY\_CONTEXT ou MQPMO\_SET\_ALL\_CONTEXT n'est pas spécifié, cette zone est ignorée en entrée et est une zone de sortie uniquement.

Une fois qu'un appel MQPUT ou MQPUT1 a abouti, cette zone contient le *UserIdentifier* qui a été transmis avec le message s'il a été inséré dans une file d'attente. Il s'agit de la valeur de *UserIdentifier* qui est conservée avec le message (voir la description de MQPMO\_RETAIN pour plus de détails sur les publications conservées) mais qui n'est pas utilisée en tant que *UserIdentifier* lorsque le message est envoyé en tant que publication aux abonnés car ils fournissent une valeur pour remplacer *UserIdentifier* dans toutes les publications qui leur sont envoyées. Si le message n'a pas de contexte, la zone est entièrement vide.

Il s'agit d'une zone de sortie pour l'appel MQGET. La longueur de cette zone est indiquée par MQ\_USER\_ID\_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 12 caractères blancs dans les autres langages de programmation.

### **AccountingToken (MQBYTE32) pour MQMD**

Il s'agit du jeton de comptabilité, qui fait partie du *contexte d'identité* du message. Pour plus d'informations sur le contexte de message, voir «MQMD-Descripteur de message», à la page 438 et Contexte de message.

AccountingToken permet à une application de facturer de manière appropriée le travail effectué à la suite du message. Le gestionnaire de files d'attente traite ces informations comme une chaîne de bits et ne vérifie pas leur contenu.

Le gestionnaire de files d'attente génère ces informations comme suit:

- Le premier octet de la zone est défini sur la longueur des informations comptables présentes dans les octets qui suivent ; cette longueur est comprise entre zéro et 30, et est stockée dans le premier octet sous la forme d'un entier binaire.
- Le second octet et les octets suivants (tels que spécifiés par la zone de longueur) sont définis sur les informations de comptabilité appropriées à l'environnement.
  - **z/OS** Sous z/OS , les informations de comptabilité sont définies sur:
    - Pour le lot z/OS , les informations de comptabilité de la carte JES JOB ou d'une instruction JES ACCT de la carte EXEC (les séparateurs de virgules sont remplacés par X'FF'). Ces informations sont tronquées, si nécessaire, à 31 octets.
    - Pour TSO, numéro de compte de l'utilisateur.
    - Pour CICS, l'identificateur d'unité de travail (UEPUOWDS) de l'unité logique 6.2 (26 octets).
    - Pour IMS, nom de bloc de spécification de programme à 8 caractères concaténé avec le jeton de récupération IMS à 16 caractères.
  - **IBM i** Sous IBM i, les informations de comptabilité sont définies sur le code de comptabilité du travail.
  - **Linux** **AIX** Sous AIX and Linux, les informations de comptabilité sont définies sur l'identificateur numérique de l'utilisateur, en caractères ASCII.
  - **Windows** Sous Windows, les informations comptables sont définies sur un identificateur de sécurité (SID) Windows dans un format compressé. Le SID identifie de manière unique l'identificateur d'utilisateur stocké dans la zone *UserIdentifier* . Lorsque le SID est stocké dans la zone *AccountingToken* , l'autorité d'identification de 6 octets (située dans le troisième octet et les suivants du SID) est omise. Par exemple, si le SID Windows a une longueur de 28 octets, 22 octets d'informations SID sont stockés dans la zone *AccountingToken* .
- Le dernier octet (octet 32) de la zone de comptabilité est défini sur le type de jeton de comptabilité (dans ce cas, MQACTT\_NT\_SECURITY\_ID, x'0b'):

#### **MQACTT\_CICS\_LUOW\_ID**

Identificateur CICS LUOW.

#### **Windows MQACTT\_NT\_SECURITY\_ID**

Identificateur de sécurité Windows .

#### **IBM i MQACTT\_OS400\_ACCOUNT\_TOKEN**

Jeton de comptabilité IBM i .

#### **UNIX MQACTT\_UNIX\_NUMERIC\_ID**

Identificateur numérique UNIX .

#### **MQACTT\_UTILISATEUR**

Jeton de comptabilité défini par l'utilisateur.

#### **MQACTT\_INCONNU**

Type de jeton de comptabilité inconnu.

Le type de jeton de comptabilité est défini sur une valeur explicite uniquement dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour IBM MQ MQI clients connecté à ces systèmes. Dans d'autres environnements, le type de jeton de comptabilité est défini sur la valeur `MQACTT_UNKNOWN`. Dans ces environnements, utilisez la zone `PutApplType` pour déduire le type de jeton de comptabilité reçu.

- Tous les autres octets sont définis sur zéro binaire.

Pour les appels `MQPUT` et `MQPUT1`, il s'agit d'une zone d'entrée-sortie si `MQPMO_SET_IDENTITY_CONTEXT` ou `MQPMO_SET_ALL_CONTEXT` est spécifié dans le paramètre **PutMsgOpts**. Si ni `MQPMO_SET_IDENTITY_CONTEXT` ni `MQPMO_SET_ALL_CONTEXT` n'est spécifié, cette zone est ignorée en entrée et est une zone de sortie uniquement. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#).

Une fois qu'un appel `MQPUT` ou `MQPUT1` a abouti, cette zone contient le `AccountingToken` qui a été transmis avec le message s'il a été inséré dans une file d'attente. Il s'agit de la valeur de `AccountingToken` qui est conservée avec le message (voir la description de `MQPMO_RETAIN` dans «Options (MQLONG) pour MQPMO», à la page 527 pour plus de détails sur les publications conservées) mais qui n'est pas utilisée comme valeur `AccountingToken` lorsque le message est envoyé en tant que publication aux abonnés car elle fournit une valeur pour remplacer `AccountingToken` dans toutes les publications qui leur sont envoyées. Si le message n'a pas de contexte, la zone est entièrement zéro binaire.

Il s'agit d'une zone de sortie pour l'appel `MQGET`.

Cette zone n'est soumise à aucune conversion basée sur le jeu de caractères du gestionnaire de files d'attente ; elle est traitée comme une chaîne de bits et non comme une chaîne de caractères.

Le gestionnaire de files d'attente ne fait rien avec les informations de cette zone. L'application doit interpréter les informations si elle souhaite les utiliser à des fins comptables.

Vous pouvez utiliser la valeur spéciale suivante pour la zone `AccountingToken` :

#### **MQACT\_AUCUN**

Aucun jeton de comptabilité n'est spécifié.

La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante `MQACT_NONE_ARRAY` est également définie ; elle a la même valeur que `MQACT_NONE`, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La longueur de cette zone est indiquée par `MQ_ACCOUNTING_TOKEN_LENGTH`. La valeur initiale de cette zone est `MQACT_NONE`.

### ***Données ApplIdentity(MQCHAR32) pour MQMD***

Il fait partie du **contexte d'identité** du message. Pour plus d'informations sur le contexte de message, voir «MQMD-Descripteur de message», à la page 438 et [Contexte de message](#).

`ApplIdentityData` est une information définie par la suite d'applications et peut être utilisée pour fournir des informations supplémentaires sur le message ou son émetteur. Le gestionnaire de files d'attente traite ces informations comme des données de type caractères, mais ne définit pas leur format. Lorsque le gestionnaire de files d'attente génère ces informations, elles sont entièrement vides.

Pour les appels `MQPUT` et `MQPUT1`, il s'agit d'une zone d'entrée-sortie si `MQPMO_SET_IDENTITY_CONTEXT` ou `MQPMO_SET_ALL_CONTEXT` est spécifié dans le paramètre **PutMsgOpts**. Si un caractère null est présent, les caractères null et les caractères suivants sont convertis en blancs par le gestionnaire de files d'attente. Si ni `MQPMO_SET_IDENTITY_CONTEXT` ni



MQPMO\_SET\_ALL\_CONTEXT n'est spécifié, cette zone est ignorée en entrée et est une zone de sortie uniquement. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#).

Une fois qu'un appel MQPUT ou MQPUT1 a abouti, cette zone contient le *ApplIdentityData* qui a été transmis avec le message s'il a été inséré dans une file d'attente. Il s'agit de la valeur de *ApplIdentityData* qui est conservée avec le message (voir la description de MQPMO\_RETAIN pour plus de détails sur les publications conservées) mais qui n'est pas utilisée en tant que *ApplIdentityData* lorsque le message est envoyé en tant que publication aux abonnés car ils fournissent une valeur pour remplacer *ApplIdentityData* dans toutes les publications qui leur sont envoyées. Si le message n'a pas de contexte, la zone est entièrement vide.

Il s'agit d'une zone de sortie pour l'appel MQGET. La longueur de cette zone est indiquée par MQ\_APPL\_IDENTITY\_DATA\_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et 32 caractères blancs dans les autres langages de programmation.

### ***PutApplType (MQLONG) pour MQMD***

Il s'agit du type d'application qui a inséré le message et qui fait partie du **contexte d'origine** du message. Pour plus d'informations sur le contexte de message, voir «MQMD-Descripteur de message», à la page 438 et [Contexte de message](#).

*PutApplType* peut avoir l'un des types standard suivants. Vous pouvez également définir vos propres types, mais uniquement avec des valeurs comprises entre MQAT\_USER\_FIRST et MQAT\_USER\_LAST.

#### **MQAT\_AIX**

Application AIX (même valeur que MQAT\_UNIX).

#### **MQAT\_AMQP**

Application de protocole AMQP

#### **MQAT\_COURTIER**

Courtier.

#### **MQAT\_CICS**

Transaction CICS .

#### **MQAT\_CICS\_BRIDGE**

CICS bridge.

#### **MQAT\_CICS\_VSE**

Transaction CICS/VSE .

#### **MQAT\_DOS**

Application IBM MQ MQI client sur PC DOS.

#### **MQAT\_DQM**

Agent de gestionnaire de files d'attente réparties.

#### **MQAT\_GARDIEN**

Application Tandem Guardian (même valeur que MQAT\_NSK).

#### **MQAT\_IMS**

Application IMS .

#### **MQAT\_IMS\_BRIDGE**

Passerelle IMS .

#### **MQAT\_JAVA**

Java.

#### **MQAT\_MVS**

Application MVS ou TSO (même valeur que MQAT\_ZOS).

#### **MQAT\_NOTES\_AGENT**

Application de l'agent Lotus Notes .

#### **MQAT\_OS390**

Application OS/390 (même valeur que MQAT\_ZOS).

**MQAT\_OS400**

Application IBM i .

**MQAT\_QMGR**

Gestionnaire de files d'attente.

**MQAT\_UNIX**

Application UNIX .

**MQAT\_VOS**

Application Stratus VOS.

**MQAT\_WINDOWS**

Application Windows 16 bits.

**MQAT\_WINDOWS\_NT**

Application Windows 32 bits.

**MQAT\_WLM**

Application du gestionnaire de charge de travail z/OS .

**MQAT\_XCF**

XCF.

**MQAT\_ZOS**

Application z/OS .

**MQAT\_PAR DEF AUT**

Type d'application par défaut.

Il s'agit du type d'application par défaut pour la plateforme sur laquelle l'application s'exécute.

**Remarque :** La valeur de cette constante est spécifique à l'environnement. Pour cette raison, compilez toujours l'application à l'aide des fichiers d'en-tête, d'inclusion ou COPY appropriés à la plateforme sur laquelle l'application s'exécutera.

**MQAT\_INCONNU**

Utilisez cette valeur pour indiquer que le type d'application est inconnu, même si d'autres informations de contexte sont présentes.

**MQAT\_USER\_FIRST**

Valeur la plus faible pour le type d'application défini par l'utilisateur.

**MQAT\_USER\_LAST**

Valeur la plus élevée pour le type d'application défini par l'utilisateur.

La valeur spéciale suivante peut également se produire:

**MQAT\_NO\_CONTEXT**

Cette valeur est définie par le gestionnaire de files d'attente lorsqu'un message est inséré sans contexte (c'est-à-dire que l'option de contexte MQPMO\_NO\_CONTEXT est spécifiée).

Lorsqu'un message est extrait, *PutApplType* peut être testé pour cette valeur afin de déterminer si le message a un contexte (il est recommandé que *PutApplType* ne soit jamais défini sur MQAT\_NO\_CONTEXT, par une application utilisant MQPMO\_SET\_ALL\_CONTEXT, si l'une des autres zones de contexte n'est pas vide).

Lorsque le gestionnaire de files d'attente génère ces informations à la suite d'une insertion d'application, la zone est définie sur une valeur déterminée par l'environnement. Sous IBM i, il est défini sur MQAT\_OS400; le gestionnaire de files d'attente n'utilise jamais MQAT\_CICS sous IBM i.

Pour les appels MQPUT et MQPUT1 , il s'agit d'une zone d'entrée-sortie si MQPMO\_SET\_ALL\_CONTEXT est spécifié dans le paramètre **PutMsgOpts** . Si MQPMO\_SET\_ALL\_CONTEXT n'est pas spécifié, cette zone est ignorée en entrée et est une zone de sortie uniquement.

Il s'agit d'une zone de sortie pour l'appel MQGET. La valeur initiale de cette zone est MQAT\_NO\_CONTEXT.

## **PutApplNom (MQCHAR28) pour MQMD**

Il s'agit du nom de l'application qui a inséré le message et qui fait partie du *contexte d'origine* du message. Le contenu diffère d'une plateforme à l'autre et peut varier d'une édition à l'autre.

Pour plus d'informations sur le contexte de message, voir «MQMD-Descripteur de message», à la page 438 et [Contexte de message](#).

Depuis IBM MQ 9.1.2, vous pouvez spécifier le nom de l'application dans des langages de programmation supplémentaires. Voir [Specifying the application name in supported programming languages](#) pour plus d'informations.

Le format de *PutApplName* dépend de la valeur de *PutApplType* et peut varier d'une édition à l'autre. Les modifications sont rares, mais elles se produisent si l'environnement change.

Lorsque le gestionnaire de files d'attente définit cette zone (c'est-à-dire, pour toutes les options à l'exception de MQPMO\_SET\_ALL\_CONTEXT), il définit la zone sur une valeur déterminée par l'environnement:

- **z/OS** Sous z/OS, le gestionnaire de files d'attente utilise:
  - Pour le lot z/OS, nom de travail à 8 caractères provenant de la carte de travail JES
  - Pour TSO, l'ID utilisateur TSO à 7 caractères
  - Pour CICS, l'ID d'application à 8 caractères, suivi de l'ID de transaction à 4 caractères
  - Pour IMS, l'identificateur système IMS à 8 caractères, suivi du nom PSB à 8 caractères
  - Pour XCF, nom de groupe XCF à 8 caractères, suivi du nom de membre XCF à 16 caractères
  - Pour un message généré par un gestionnaire de files d'attente, les 28 premiers caractères du nom du gestionnaire de files d'attente
  - Pour la mise en file d'attente répartie sans CICS, nom de travail à 8 caractères de l'initiateur de canal, suivi du nom à 8 caractères du module placé dans la file d'attente de rebut, suivi d'un identificateur de tâche à 8 caractères.

Le ou les noms sont chacun complétés à droite par des blancs, comme n'importe quel espace dans le reste de la zone. Lorsqu'il y a plusieurs noms, il n'y a pas de séparateur entre eux.

- **Windows** Sur les systèmes Windows, le gestionnaire de files d'attente utilise les noms suivants:
  - Pour une application CICS, le nom de transaction CICS
  - Pour une application nonCICS, les 28 caractères les plus à droite du nom qualifié complet de l'exécutable
- **IBM i** Sous IBM i, le gestionnaire de files d'attente utilise le nom de travail qualifié complet.
- **Linux** **AIX** Sous AIX and Linux, le gestionnaire de files d'attente utilise les noms suivants:
  - Pour une application CICS, le nom de transaction CICS
  - Pour une application nonCICS, MQ demande le nom du processus au système d'exploitation. Ce nom est renvoyé en tant que nom de fichier programme, sans chemin d'accès complet. Ensuite, MQ place ce nom de processus dans MQMD.PutApplName de PutApplName se présente comme suit:

### **AIX AIX**

Si le nom est inférieur ou égal à 28 octets, le nom est inséré, complété à droite par des espaces.

Si le nom est supérieur à 28 octets, les 28 octets les plus à gauche du nom sont insérés.

### **Linux Linux**

Si le nom est inférieur ou égal à 15 octets, le nom est inséré, complété à droite par des espaces.

Si le nom est supérieur à 15 octets, les 15 octets les plus à gauche du nom sont insérés, complétés à droite par des espaces.

Par exemple, si vous exécutez `/opt/mqm/samp/bin/amqsput QNAME QMNAME`, le nom PutAppl est 'amqsput'. Cette zone MQCHAR28 contient 21 caractères d'espace de remplissage. Notez que le chemin d'accès complet incluant `/opt/mqm/samp/bin` n'est pas inclus dans le nom PutAppl.

Pour les appels MQPUT et MQPUT1, il s'agit d'une zone d'entrée-sortie si MQPMO\_SET\_ALL\_CONTEXT est spécifié dans le paramètre **PutMsgOpts**. Toutes les informations qui suivent un caractère null dans la zone sont supprimées. Le caractère null et tous les caractères suivants sont convertis en blancs par le gestionnaire de files d'attente. Si MQPMO\_SET\_ALL\_CONTEXT n'est pas spécifié, cette zone est ignorée en entrée et est une zone de sortie uniquement.

### ***PutDate (MQCHAR8) pour MQMD***

Il s'agit de la date à laquelle le message a été inséré et qui fait partie du **contexte d'origine** du message. Pour plus d'informations sur le contexte de message, voir «MQMD-Descripteur de message», à la page 438 et [Contexte de message](#).

Le format utilisé pour la date à laquelle cette zone est générée par le gestionnaire de files d'attente est le suivant:

- AAAAMMJJ

où les caractères représentent:

#### **AAAA**

année (quatre chiffres)

#### **MM**

mois de l'année (01 à 12)

#### **JJ**

jour du mois (01 à 31)

Le temps moyen de Greenwich (GMT) est utilisé pour les zones *PutDate* et *PutTime*, à condition que l'horloge système soit définie avec précision sur le temps moyen de Greenwich.

Si le message a été inséré dans une unité de travail, la date est celle à laquelle le message a été inséré et non celle à laquelle l'unité de travail a été validée.

Pour les appels MQPUT et MQPUT1, il s'agit d'une zone d'entrée-sortie si MQPMO\_SET\_ALL\_CONTEXT est spécifié dans le paramètre **PutMsgOpts**. Le contenu de la zone n'est pas vérifié par le gestionnaire de files d'attente, sauf que les informations qui suivent un caractère null dans la zone sont supprimées. Le gestionnaire de files d'attente convertit le caractère null et les caractères suivants en blancs. Si MQPMO\_SET\_ALL\_CONTEXT n'est pas spécifié, cette zone est ignorée en entrée et est une zone de sortie uniquement.

Il s'agit d'une zone de sortie pour l'appel MQGET. La longueur de cette zone est indiquée par MQ\_PUT\_DATE\_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et 8 caractères blancs dans les autres langages de programmation.

### ***PutTime (MQCHAR8) pour MQMD***

Il s'agit de l'heure à laquelle le message a été inséré et il fait partie du **contexte d'origine** du message. Pour plus d'informations sur le contexte de message, voir «MQMD-Descripteur de message», à la page 438 et [Contexte de message](#).

Le format utilisé pour l'heure à laquelle cette zone est générée par le gestionnaire de files d'attente est le suivant:

- HHMMSSSTH

où les caractères représentent (dans l'ordre):

#### **HH**

heures (00 à 23)

#### **MM**

minutes (00 à 59)

**SS**

secondes (00 à 59 ; voir remarque)

**T**

dixièmes de seconde (0 à 9)

**H**

centièmes de seconde (0 à 9)

**Remarque :** Si l'horloge système est synchronisée selon une norme de temps très précise, il est possible en de rares occasions que 60 ou 61 soient renvoyés pour les secondes dans *PutTime*. Cela se produit lorsque des secondes bissextiles sont insérées dans la norme de temps globale.

Le temps moyen de Greenwich (GMT) est utilisé pour les zones *PutDate* et *PutTime*, à condition que l'horloge système soit définie avec précision sur le temps moyen de Greenwich.

Si le message a été inséré dans une unité de travail, l'heure est celle à laquelle le message a été inséré et non celle à laquelle l'unité de travail a été validée.

Pour les appels MQPUT et MQPUT1, il s'agit d'une zone d'entrée-sortie si MQPMO\_SET\_ALL\_CONTEXT est spécifié dans le paramètre **PutMsgOpts**. Le gestionnaire de files d'attente ne vérifie pas le contenu de la zone, sauf que les informations qui suivent un caractère null dans la zone sont supprimées. Le gestionnaire de files d'attente convertit le caractère null et les caractères suivants en blancs. Si MQPMO\_SET\_ALL\_CONTEXT n'est pas spécifié, cette zone est ignorée en entrée et est une zone de sortie uniquement.

Il s'agit d'une zone de sortie pour l'appel MQGET. La longueur de cette zone est indiquée par MQ\_PUT\_TIME\_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et 8 caractères blancs dans les autres langages de programmation.

**Données ApplOrigin(MQCHAR4) pour MQMD**

Il fait partie du *contexte d'origine* du message. Pour plus d'informations sur le contexte de message, voir «MQMD-Descripteur de message», à la page 438 et [Contexte de message](#).

ApplOriginData est une information définie par la suite d'applications qui peut être utilisée pour fournir des informations supplémentaires sur l'origine du message. Par exemple, elle peut être définie par des applications s'exécutant avec des droits utilisateur appropriés pour indiquer si les données d'identité sont dignes de confiance.

Le gestionnaire de files d'attente traite ces informations comme des données de type caractères, mais ne définit pas leur format. Lorsque le gestionnaire de files d'attente génère ces informations, elles sont entièrement vides.

Pour les appels MQPUT et MQPUT1, il s'agit d'une zone d'entrée-sortie si MQPMO\_SET\_ALL\_CONTEXT est spécifié dans le paramètre **PutMsgOpts**. Toutes les informations qui suivent un caractère null dans la zone sont supprimées. Le gestionnaire de files d'attente convertit le caractère null et les caractères suivants en blancs. Si MQPMO\_SET\_ALL\_CONTEXT n'est pas spécifié, cette zone est ignorée en entrée et est une zone de sortie uniquement.

Il s'agit d'une zone de sortie pour l'appel MQGET. La longueur de cette zone est indiquée par MQ\_APPL\_ORIGIN\_DATA\_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et 4 caractères blancs dans les autres langages de programmation.

Lorsque le message est publié, bien que ApplOriginData soit défini, il est vide dans l'abonnement qu'il reçoit.

**GroupId (MQBYTE24) pour MQMD**

Il s'agit d'une chaîne d'octets utilisée pour identifier le groupe de messages particulier ou le message logique auquel appartient le message physique. *GroupId* est également utilisé si la segmentation est autorisée pour le message. Dans tous ces cas, *GroupId* a une valeur non nulle et un ou plusieurs des indicateurs suivants sont définis dans la zone *MsgFlags* :

- GROUPE MQMF\_MSG\_IN\_GROUP

- MQMF\_LAST\_MSG\_IN\_GROUP
- SEGMENT\_MQMF
- SEGMENT\_DERNIER\_MQMF\_SEGMENT
- MQMF\_SEGMENTATION\_ALLOWED

Si aucun de ces indicateurs n'est défini, *GroupId* a la valeur null spéciale MQGI\_NONE.

L'application n'a pas besoin de définir cette zone dans l'appel MQPUT ou MQGET si:

- Dans l'appel MQPUT, MQPMO\_LOGICAL\_ORDER est spécifié.
- Dans l'appel MQGET, MQMO\_MATCH\_GROUP\_ID n'est pas spécifié.

Il s'agit des méthodes recommandées pour utiliser ces appels pour les messages qui ne sont pas des messages de rapport. Toutefois, si l'application requiert davantage de contrôle ou si l'appel est MQPUT1, l'application doit s'assurer que *GroupId* est défini sur une valeur appropriée.

Les groupes de messages et les segments ne peuvent être traités correctement que si l'identificateur de groupe est unique. Pour cette raison, les applications *ne doivent pas générer leurs propres identificateurs de groupe* ; à la place, les applications doivent effectuer l'une des opérations suivantes:

- Si MQPMO\_LOGICAL\_ORDER est spécifié, le gestionnaire de files d'attente génère automatiquement un identificateur de groupe unique pour le premier message du groupe ou du segment du message logique et utilise cet identificateur de groupe pour les messages restants du groupe ou des segments du message logique, de sorte que l'application n'a pas besoin d'effectuer d'action spéciale. Il s'agit de la procédure recommandée.
- Si MQPMO\_LOGICAL\_ORDER n'est pas spécifié, l'application doit demander au gestionnaire de files d'attente de générer l'identificateur de groupe, en définissant *GroupId* sur MQGI\_NONE sur le premier appel MQPUT ou MQPUT1 pour un message dans le groupe ou le segment du message logique. L'identificateur de groupe renvoyé par le gestionnaire de files d'attente en sortie de cet appel doit ensuite être utilisé pour les messages restants du groupe ou des segments du message logique. Si un groupe de messages contient des messages segmentés, le même identificateur de groupe doit être utilisé pour tous les segments et messages du groupe.

Lorsque MQPMO\_LOGICAL\_ORDER n'est pas spécifié, les messages des groupes et les segments de messages logiques peuvent être placés dans n'importe quel ordre (par exemple, dans l'ordre inverse), mais l'identificateur de groupe doit être alloué par le *premier* appel MQPUT ou MQPUT1 émis pour l'un de ces messages.

Lors de l'entrée dans les appels MQPUT et MQPUT1, le gestionnaire de files d'attente utilise la valeur décrite dans Ordre physique dans une file d'attente. Dans la sortie des appels MQPUT et MQPUT1, le gestionnaire de files d'attente définit cette zone sur la valeur qui a été envoyée avec le message si l'objet ouvert est une file d'attente unique et non une liste de distribution, mais la laisse inchangée si l'objet ouvert est une liste de distribution. Dans ce dernier cas, si l'application doit connaître les identificateurs de groupe générés, elle doit fournir des enregistrements MQPMR contenant la zone *GroupId*.

En entrée de l'appel MQGET, le gestionnaire de files d'attente utilise la valeur décrite dans Tableau 495, à la page 409. Dans la sortie de l'appel MQGET, le gestionnaire de files d'attente définit cette zone sur la valeur du message extrait.

La valeur spéciale suivante est définie:

#### **MQGI\_AUCUN**

Aucun identificateur de groupe n'a été indiqué.

La valeur est zéro binaire pour la longueur de la zone. Il s'agit de la valeur utilisée pour les messages qui ne sont pas dans des groupes, qui ne sont pas des segments de messages logiques et pour lesquels la segmentation n'est pas autorisée.

Pour le langage de programmation C, la constante MQGI\_NONE\_ARRAY est également définie ; elle a la même valeur que MQGI\_NONE, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

La longueur de cette zone est indiquée par MQ\_GROUP\_ID\_LENGTH. La valeur initiale de cette zone est MQGI\_NONE. Cette zone est ignorée si *Version* est inférieur à MQMD\_VERSION\_2.

### **MsgSeqNuméro (MQLONG) pour MQMD**

Il s'agit du numéro de séquence d'un message logique au sein d'un groupe.

Les numéros de séquence commencent à 1 et augmentent de 1 pour chaque nouveau message logique du groupe, jusqu'à un maximum de 999 999 999. Un message physique ne se trouvant pas dans un groupe possède le numéro de séquence 1.

L'application n'a pas besoin de définir cette zone dans l'appel MQPUT ou MQGET si:

- Dans l'appel MQPUT, MQPMO\_LOGICAL\_ORDER est spécifié.
- Dans l'appel MQGET, MQMO\_MATCH\_MSG\_SEQ\_NUMBER n'est pas spécifié.

Il s'agit des méthodes recommandées pour utiliser ces appels pour les messages qui ne sont pas des messages de rapport. Toutefois, si l'application requiert davantage de contrôle ou si l'appel est MQPUT1, l'application doit s'assurer que *MsgSeqNumber* est défini sur une valeur appropriée.

Lors de l'entrée dans les appels MQPUT et MQPUT1, le gestionnaire de files d'attente utilise la valeur décrite dans Ordre physique dans une file d'attente. Dans la sortie des appels MQPUT et MQPUT1, le gestionnaire de files d'attente définit cette zone sur la valeur qui a été envoyée avec le message.

En entrée de l'appel MQGET, le gestionnaire de files d'attente utilise la valeur indiquée dans Tableau 495, à la page 409. Dans la sortie de l'appel MQGET, le gestionnaire de files d'attente définit cette zone sur la valeur du message extrait.

La valeur initiale de cette zone est 1. Cette zone est ignorée si *Version* est inférieur à MQMD\_VERSION\_2.

### **Décalage (MQLONG) pour MQMD**

Il s'agit du décalage en octets des données du message physique à partir du début du message logique dont font partie les données. Ces données sont appelées un *segment*. Le décalage est compris entre 0 et 999 999 999. Un message physique qui n'est pas un segment d'un message logique a un décalage de zéro.

L'application n'a pas besoin de définir cette zone dans l'appel MQPUT ou MQGET si:

- Dans l'appel MQPUT, MQPMO\_LOGICAL\_ORDER est spécifié.
- Dans l'appel MQGET, MQMO\_MATCH\_OFFSET n'est pas spécifié.

Il s'agit des méthodes recommandées pour utiliser ces appels pour les messages qui ne sont pas des messages de rapport. Toutefois, si l'application ne respecte pas ces conditions ou si l'appel est MQPUT1, l'application doit s'assurer que *Offset* est défini sur une valeur appropriée.

Lors de l'entrée dans les appels MQPUT et MQPUT1, le gestionnaire de files d'attente utilise la valeur décrite dans Ordre physique dans une file d'attente. Dans la sortie des appels MQPUT et MQPUT1, le gestionnaire de files d'attente définit cette zone sur la valeur qui a été envoyée avec le message.

Pour un message de rapport signalant un segment d'un message logique, la zone *OriginalLength* (à condition qu'elle ne soit pas MQOL\_UNDEFINED) est utilisée pour mettre à jour le décalage dans les informations de segment conservées par le gestionnaire de files d'attente.

En entrée de l'appel MQGET, le gestionnaire de files d'attente utilise la valeur indiquée dans Tableau 495, à la page 409. Dans la sortie de l'appel MQGET, le gestionnaire de files d'attente définit cette zone sur la valeur du message extrait.

La valeur initiale de cette zone est zéro. Cette zone est ignorée si *Version* est inférieur à MQMD\_VERSION\_2.

### **MsgFlags (MQLONG) pour MQMD**

Les MsgFlags sont des indicateurs qui spécifient les attributs du message ou qui contrôlent son traitement.

Les indicateurs de message (MsgFlags) se répartissent dans les catégories suivantes:

- Indicateurs de segmentation

- Indicateurs de statut

**Indicateurs de segmentation:** Lorsqu'un message est trop volumineux pour une file d'attente, une tentative d'insertion du message dans la file d'attente échoue généralement. La segmentation est une technique par laquelle le gestionnaire de files d'attente ou l'application fractionne le message en éléments plus petits appelés segments et place chaque segment dans la file d'attente sous la forme d'un message physique distinct. L'application qui extrait le message peut soit extraire les segments un par un, soit demander au gestionnaire de files d'attente de réassembler les segments en un seul message renvoyé par l'appel MQGET. Ce dernier est obtenu en spécifiant l'option MQGMO\_COMPLETE\_MSG dans l'appel MQGET et en fournissant une mémoire tampon suffisamment grande pour contenir le message complet. (Voir «Options MQGMO-Get-message», à la page 382 pour plus de détails sur l'option MQGMO\_COMPLETE\_MSG.) Un message peut être segmenté au niveau du gestionnaire de files d'attente d'envoi, au niveau d'un gestionnaire de files d'attente intermédiaire ou au niveau du gestionnaire de files d'attente de destination.

Vous pouvez spécifier l'une des options suivantes pour contrôler la segmentation d'un message:

#### **MQMF\_SEGMENTATION\_INHIBÉE**

Cette option empêche le message d'être divisé en segments par le gestionnaire de files d'attente. Si elle est spécifiée pour un message qui est déjà un segment, cette option empêche le segment d'être divisé en segments plus petits.

La valeur de cet indicateur est zéro binaire. Il s'agit de l'option par défaut.

#### **MQMF\_SEGMENTATION\_ALLOWED**

Cette option permet au message d'être divisé en segments par le gestionnaire de files d'attente. Si elle est spécifiée pour un message qui est déjà un segment, cette option permet de le scinder en segments plus petits. MQMF\_SEGMENTATION\_ALLOWED peut être défini sans que MQMF\_SEGMENT ou MQMF\_LAST\_SEGMENT ne soit défini.

- Sous z/OS, le gestionnaire de files d'attente ne prend pas en charge la segmentation des messages. Si un message est trop volumineux pour la file d'attente, l'appel MQPUT ou MQPUT1 échoue avec le code anomalie MQRC\_MSG\_TOO\_BIG\_FOR\_Q. Toutefois, l'option MQMF\_SEGMENTATION\_ALLOWED peut toujours être spécifiée et permet de segmenter le message au niveau d'un gestionnaire de files d'attente éloignées.

Lorsque le gestionnaire de files d'attente segmente un message, il active l'indicateur MQMF\_SEGMENT dans la copie du MQMD envoyé avec chaque segment, mais ne modifie pas les paramètres de ces indicateurs dans le MQMD fourni par l'application sur l'appel MQPUT ou MQPUT1. Pour le dernier segment du message logique, le gestionnaire de files d'attente active également l'indicateur MQMF\_LAST\_SEGMENT dans le MQMD envoyé avec le segment.

**Remarque :** Faites attention lors de l'insertion de messages avec MQMF\_SEGMENTATION\_ALLOWED mais sans MQPMO\_LOGICAL\_ORDER. Si le message est:

- N'est pas un segment, et
- Pas dans un groupe, et
- Non réacheminé,

l'application doit réinitialiser la zone *GroupId* sur MQGI\_NONE avant chaque appel MQPUT ou MQPUT1, afin que le gestionnaire de files d'attente puisse générer un identificateur de groupe unique pour chaque message. Si tel n'est pas le cas, les messages non liés peuvent avoir le même identificateur de groupe, ce qui peut entraîner un traitement incorrect par la suite. Pour plus d'informations sur la réinitialisation de la zone *GroupId*, voir les descriptions de la zone *GroupId* et de l'option MQPMO\_LOGICAL\_ORDER.

Le gestionnaire de files d'attente divise les messages en segments selon les besoins afin que les segments (plus les données d'en-tête requises) tiennent dans la file d'attente. Toutefois, il existe une limite inférieure pour la taille d'un segment généré par le gestionnaire de files d'attente et seul le dernier segment créé à partir d'un message peut être inférieur à cette limite (la limite inférieure pour la taille d'un segment généré par une application est d'un octet). Les segments générés par le



gestionnaire de files d'attente peuvent être de longueur inégale. Le gestionnaire de files d'attente traite le message comme suit:

- Les formats définis par l'utilisateur sont divisés sur des limites multiples de 16 octets ; le gestionnaire de files d'attente ne génère pas de segments inférieurs à 16 octets (autres que le dernier segment).
- Les formats intégrés autres que MQFMT\_STRING sont fractionnés aux points appropriés à la nature des données présentes. Toutefois, le gestionnaire de files d'attente ne fractionne jamais un message au milieu d'une structure d'en-tête IBM MQ . Cela signifie qu'un segment contenant une structure d'en-tête MQ unique ne peut pas être fractionné davantage par le gestionnaire de files d'attente et que, par conséquent, la taille de segment minimale possible pour ce message est supérieure à 16 octets.

Le deuxième segment ou un segment ultérieur généré par le gestionnaire de files d'attente commence par l'un des segments suivants:

- Une structure d'en-tête MQ
  - Début des données de message d'application
  - Une partie de l'utilisation des données de message d'application
- MQFMT\_STRING est divisé sans tenir compte de la nature des données présentes (SBCS, DBCS ou SBCS/DBCS mixte). Lorsque la chaîne est de type DBCS ou SBCS/DBCS mixte, il se peut que des segments ne puissent pas être convertis d'un jeu de caractères à un autre. Le gestionnaire de files d'attente ne fractionne jamais les messages MQFMT\_STRING en segments de moins de 16 octets (autres que le dernier segment).
  - Le gestionnaire de files d'attente définit les zones *Format*, *CodedCharSetIdet Encoding* dans le MQMD de chaque segment pour décrire correctement les données présentes au *début* du segment ; le nom de format est soit le nom d'un format intégré, soit le nom d'un format défini par l'utilisateur.
  - La zone *Report* dans le MQMD des segments dont le *Offset* est supérieur à zéro est modifiée. Pour chaque type de rapport, si l'option de rapport est MQRO\_\*\_WITH\_DATA, mais que le segment ne peut pas contenir les 100 premiers octets de données utilisateur (c'est-à-dire les données qui suivent les structures d'en-tête IBM MQ éventuellement présentes), l'option de rapport est remplacée par MQRO\_\*.

Le gestionnaire de files d'attente respecte les règles ci-dessus, mais fractionne les messages de manière imprévisible ; ne faites pas d'hypothèses sur l'endroit où un message est fractionné.

Pour les messages *persistants* , le gestionnaire de files d'attente peut effectuer une segmentation uniquement dans une unité d'oeuvre:

- Si l'appel MQPUT ou MQPUT1 s'exécute dans une unité de travail définie par l'utilisateur, cette unité de travail est utilisée. Si l'appel échoue lors du processus de segmentation, le gestionnaire de files d'attente supprime tous les segments placés dans la file d'attente suite à l'échec de l'appel. Toutefois, l'échec n'empêche pas la validation de l'unité de travail.
- Si l'appel est effectué en dehors d'une unité de travail définie par l'utilisateur et qu'il n'existe aucune unité de travail définie par l'utilisateur, le gestionnaire de files d'attente crée une unité de travail uniquement pour la durée de l'appel. Si l'appel aboutit, le gestionnaire de files d'attente valide automatiquement l'unité de travail. Si l'appel échoue, le gestionnaire de files d'attente annule l'unité d'oeuvre.
- Si l'appel est exécuté en dehors d'une unité de travail définie par l'utilisateur, mais qu'une unité de travail définie par l'utilisateur existe, le gestionnaire de files d'attente ne peut pas effectuer de segmentation. Si le message ne nécessite pas de segmentation, l'appel peut tout de même aboutir. Mais si le message requiert une segmentation, l'appel échoue avec le code anomalie MQRC\_UOW\_NOT\_AVAILABLE.

Pour les messages *non persistants* , le gestionnaire de files d'attente ne requiert pas qu'une unité d'oeuvre soit disponible pour effectuer la segmentation.

Soyez particulièrement prudent lorsque vous convertissez des données dans des messages qui peuvent être segmentés:

- Si l'application de réception convertit des données dans l'appel MQGET et spécifie l'option MQGMO\_COMPLETE\_MSG, l'exit de conversion de données reçoit le message complet de l'exit à convertir et le fait que le message a été segmenté est visible pour l'exit.
- Si l'application de réception extrait un segment à la fois, l'exit de conversion de données est appelé pour convertir un segment à la fois. L'exit doit donc convertir les données dans un segment indépendamment des données dans n'importe lequel des autres segments.

Si la nature des données dans le message est telle que la segmentation arbitraire des données sur des limites de 16 octets peut entraîner des segments qui ne peuvent pas être convertis par l'exit, ou si le format est MQFMT\_STRING et que le jeu de caractères est DBCS ou SBCS/DBCS mixte, l'application émettrice doit créer et placer les segments, en spécifiant MQMF\_SEGMENTATION\_INHIBÉE pour supprimer la segmentation supplémentaire. De cette manière, l'application émettrice peut s'assurer que chaque segment contient suffisamment d'informations pour permettre à l'exit de conversion de données de convertir le segment avec succès.

- Si la conversion de l'émetteur est indiquée pour un agent MCA, ce dernier convertit uniquement les messages qui ne sont pas des segments de messages logiques ; l'agent MCA ne tente jamais de convertir les messages qui sont des segments.

Cet indicateur est un indicateur d'entrée sur les appels MQPUT et MQPUT1 et un indicateur de sortie sur l'appel MQGET. Lors de ce dernier appel, le gestionnaire de files d'attente renvoie également la valeur de l'indicateur à la zone *Segmentation* dans MQGMO.

La valeur initiale de cet indicateur est MQMF\_SEGMENTATION\_INHIBÉE.

**Indicateurs de statut:** indicateurs qui indiquent si le message physique appartient à un groupe de messages, s'il s'agit d'un segment d'un message logique, les deux ou aucun des deux. Un ou plusieurs des éléments suivants peuvent être spécifiés dans l'appel MQPUT ou MQPUT1 ou renvoyés par l'appel MQGET:

#### **GROUPE MQMF\_MSG\_IN\_GROUP**

Le message est membre d'un groupe.

#### **MQMF\_LAST\_MSG\_IN\_GROUP**

Le message est le dernier message logique d'un groupe.

Si cet indicateur est défini, le gestionnaire de files d'attente active MQMF\_MSG\_IN\_GROUP dans la copie de MQMD envoyée avec le message, mais ne modifie pas les paramètres de ces indicateurs dans le MQMD fourni par l'application sur l'appel MQPUT ou MQPUT1 .

Un groupe ne peut comporter qu'un seul message logique. Si tel est le cas, MQMF\_LAST\_MSG\_IN\_GROUP est défini, mais la zone *MsgSeqNumber* a la valeur 1.

#### **SEGMENT\_MQMF**

Le message est un segment d'un message logique.

Lorsque MQMF\_SEGMENT est spécifié sans MQMF\_LAST\_SEGMENT, la longueur des données de message d'application dans le segment ( à l'exclusion des longueurs des structures d'en-tête IBM MQ pouvant être présentes) doit être au moins une. Si la longueur est égale à zéro, l'appel MQPUT ou MQPUT1 échoue avec le code anomalie MQRC\_SEGMENT\_LENGTH\_ZERO.

Sous z/OS, cette option n'est pas prise en charge si le message est inséré dans une file d'attente dont le type d'index est MQIT\_GROUP\_ID.

#### **SEGMENT\_DERNIER\_MQMF\_SEGMENT**

Le message est le dernier segment d'un message logique.

Si cet indicateur est défini, le gestionnaire de files d'attente active MQMF\_SEGMENT dans la copie de MQMD envoyée avec le message, mais ne modifie pas les paramètres de ces indicateurs dans le MQMD fourni par l'application sur l'appel MQPUT ou MQPUT1 .

Un message logique ne peut se composer que d'un seul segment. Si tel est le cas, MQMF\_LAST\_SEGMENT est défini, mais la zone *Offset* a la valeur zéro.

Lorsque MQMF\_LAST\_SEGMENT est spécifié, la longueur des données de message d'application dans le segment ( à l'exclusion des longueurs des structures d'en-tête éventuellement présentes) peut être égale à zéro.

Sous z/OS, cette option n'est pas prise en charge si le message est inséré dans une file d'attente dont le type d'index est MQIT\_GROUP\_ID.

L'application doit s'assurer que ces indicateurs sont correctement définis lors de l'insertion de messages. Si MQPMO\_LOGICAL\_ORDER est spécifié ou a été spécifié dans l'appel MQPUT précédent pour l'identificateur de file d'attente, les paramètres des indicateurs doivent être cohérents avec les informations de groupe et de segment conservées par le gestionnaire de files d'attente pour l'identificateur de file d'attente. Les conditions suivantes s'appliquent aux appels MQPUT *successifs* pour le descripteur de file d'attente lorsque MQPMO\_LOGICAL\_ORDER est spécifié:

- S'il n'y a pas de groupe ou de message logique en cours, tous ces indicateurs (et leurs combinaisons) sont valides.
- Une fois que MQMF\_MSG\_IN\_GROUP a été spécifié, il doit rester en fonction jusqu'à ce que MQMF\_LAST\_MSG\_IN\_GROUP soit spécifié. L'appel échoue avec le code anomalie MQRC\_INCOMPLETE\_GROUP si cette condition n'est pas satisfaite.
- Une fois que MQMF\_SEGMENT a été spécifié, il doit rester en fonction jusqu'à ce que MQMF\_LAST\_SEGMENT soit spécifié. L'appel échoue avec le code anomalie MQRC\_INCOMPLETE\_MSG si cette condition n'est pas satisfaite.
- Une fois que MQMF\_SEGMENT a été spécifié sans MQMF\_MSG\_IN\_GROUP, MQMF\_MSG\_IN\_GROUP doit rester *désactivé* jusqu'à ce que MQMF\_LAST\_SEGMENT ait été spécifié. L'appel échoue avec le code anomalie MQRC\_INCOMPLETE\_MSG si cette condition n'est pas satisfaite.

Ordre physique dans une file d'attente affiche les combinaisons valides des indicateurs et les valeurs utilisées pour les différentes zones.

Ces indicateurs sont des indicateurs d'entrée sur les appels MQPUT et MQPUT1 et des indicateurs de sortie sur l'appel MQGET. Lors de ce dernier appel, le gestionnaire de files d'attente répercute également les valeurs des indicateurs dans les zones *GroupStatus* et *SegmentStatus* de MQGMO.

Vous ne pouvez pas utiliser de messages groupés ou segmentés avec la fonction de publication / abonnement.

**Indicateurs par défaut:** Vous pouvez indiquer ce qui suit pour indiquer que le message possède des attributs par défaut:

#### **MQMF\_AUCUN**

Aucun indicateur de message (attributs de message par défaut).

Cela inhibe la segmentation et indique que le message ne fait pas partie d'un groupe et n'est pas un segment d'un message logique. MQMF\_NONE est défini pour faciliter la documentation du programme. Il n'est pas prévu que cet indicateur soit utilisé avec d'autres, mais comme sa valeur est nulle, une telle utilisation ne peut pas être détectée.

La zone *MsgFlags* est partitionnée en sous-zones ; pour plus de détails, voir «Options de rapport et indicateurs de message», à la page 947.

La valeur initiale de cette zone est MQMF\_NONE. Cette zone est ignorée si *Version* est inférieur à MQMD\_VERSION\_2.

#### **OriginalLength (MQLONG) pour MQMD**

Cette zone est pertinente uniquement pour les messages de rapport qui sont des segments. Il indique la longueur du segment de message auquel se rapporte le message de rapport ; il ne précise pas la longueur du message logique dont fait partie le segment, ni la longueur des données du message de rapport.

**Remarque :** Lors de la génération d'un message de rapport pour un message qui est un segment, le gestionnaire de files d'attente et l'agent de canal de message sont copiés dans le MQMD pour le message de rapport dans les zones *GroupId*, *MsgSeqNumber*, *Offset* et *MsgFlags* du message d'origine. Par

conséquent, le message de rapport est également un segment. Les applications qui génèrent des messages de rapport doivent faire de même et définir correctement la zone *OriginalLength*.

La valeur spéciale suivante est définie:

### **MQOL\_UNDEFINI**

Longueur d'origine du message non définie.

*OriginalLength* est une zone d'entrée dans les appels MQPUT et MQPUT1, mais la valeur fournie par l'application n'est acceptée que dans des circonstances particulières:

- Si le message en cours d'insertion est un segment et est également un message de rapport, le gestionnaire de files d'attente accepte la valeur spécifiée. La valeur doit être:
  - Supérieur à zéro si le segment n'est pas le dernier segment
  - Pas moins de zéro si le segment est le dernier segment
  - Pas moins que la longueur des données présentes dans le message

Si ces conditions ne sont pas satisfaites, l'appel échoue avec le code anomalie MQRC\_ORIGINAL\_LENGTH\_ERROR.

- Si le message en cours d'insertion est un segment mais pas un message de rapport, le gestionnaire de files d'attente ignore la zone et utilise à la place la longueur des données de message d'application.
- Dans tous les autres cas, le gestionnaire de files d'attente ignore la zone et utilise la valeur MQOL\_UNDEFINED à la place.

Il s'agit d'une zone de sortie de l'appel MQGET.

La valeur initiale de cette zone est MQOL\_UNDEFINED. Cette zone est ignorée si *Version* est inférieur à MQMD\_VERSION\_2.

## **MQMDE-Extension de descripteur de message**

La structure MQMDE décrit les données qui précèdent parfois les données de message d'application. La structure contient les zones MQMD qui existent dans le MQMD version-2, mais pas dans le MQMD version-1.

## **Disponibilité**

Tous les systèmes IBM MQ, plus IBM MQ MQI clients connectés à ces systèmes.

## **Format**

MQFMT\_MD\_EXTENSION

## **Jeu de caractères et codage**

Les données de MQMDE doivent être dans le jeu de caractères et le codage du gestionnaire de files d'attente local; ces données sont fournies par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et MQENC\_NATIVE pour le langage de programmation C.

Définissez le jeu de caractères et le codage de MQMDE dans les zones *CodedCharSetId* et *Encoding* dans:

- MQMD (si la structure MQMDE est au début des données de message), ou
- Structure d'en-tête qui précède la structure MQMDE (tous les autres cas).

Si le MQMDE ne fait pas partie du jeu de caractères et du codage du gestionnaire de files d'attente, il est accepté mais non respecté, c'est-à-dire que le MQMDE est traité comme des données de message.

**Remarque :** Sous Windows, les applications compilées avec Micro Focus COBOL utilisent une valeur de MQENC\_NATIVE différente du codage du gestionnaire de files d'attente. Bien que les zones numériques de la structure MQMD sur les appels MQPUT, MQPUT1 et MQGET doivent être dans le codage Micro Focus

COBOL, les zones numériques de la structure MQMDE doivent être dans le codage du gestionnaire de files d'attente. Ce dernier est donné par MQENC\_NATIVE pour le langage de programmation C et a la valeur 546.

## Utilisation

Les applications qui utilisent un MQMD version-2 ne seront pas confrontées à une structure MQMDE. Toutefois, des applications spécialisées et des applications qui continuent d'utiliser un MQMD version-1 peuvent rencontrer un MQMDE dans certaines situations. La structure MQMDE peut se produire dans les cas suivants:

- Spécifié sur les appels MQPUT et MQPUT1
- Renvoyé par l'appel MQGET
- Dans les messages des files d'attente de transmission

## MQMDE spécifié sur les appels MQPUT et MQPUT1

Dans les appels MQPUT et MQPUT1, si l'application fournit un MQMD version-1, l'application peut éventuellement préfixer les données de message avec un MQMDE, en définissant la zone *Format* de MQMD sur MQFMT\_MD\_EXTENSION pour indiquer qu'un MQMDE est présent. Si l'application ne fournit pas de MQMDE, le gestionnaire de files d'attente utilise les valeurs par défaut pour les zones du MQMDE. Les valeurs par défaut utilisées par le gestionnaire de files d'attente sont les mêmes que les valeurs initiales de la structure ; voir [Tableau 503](#), à la page 494.

Si l'application fournit un version-2 MQMD et préfixe les données de message d'application avec un MQMDE, les structures sont traitées comme indiqué dans le tableau suivant.

<i>Tableau 502. Action du gestionnaire de files d'attente lorsque MQMDE est spécifié sur MQPUT ou MQPUT1 pour MQMDE</i>			
Version MQMD	Valeurs des zones version-2	Valeurs des zones correspondantes dans MQMDE	Action effectuée par le gestionnaire de files d'attente
1	-	Valide	MQMDE est honoré
2	Par défaut	Valide	MQMDE est honoré
2	Pas de valeur par défaut	Valide	MQMDE est traité comme des données de message
1 ou 2	Tous	Non valide	L'appel échoue avec un code anomalie approprié
1 ou 2	Tous	MQMDE est dans un jeu de caractères ou un codage incorrect, ou est une version non prise en charge	MQMDE est traité comme des données de message
<b>Remarque :</b> Sous z/OS, si l'application spécifie un MQMD version-1 avec un MQMDE, le gestionnaire de files d'attente valide le MQMDE uniquement si la file d'attente possède un <i>IndexType</i> d'ID MQIT_GROUP_ID.			

Il y a un cas particulier. Si l'application utilise un MQMD version-2 pour insérer un message qui est un segment (c'est-à-dire que l'indicateur MQMF\_SEGMENT ou MQMF\_LAST\_SEGMENT est défini) et que le nom de format dans le MQMD est MQFMT\_DEAD\_LETTER\_HEADER, le gestionnaire de files d'attente génère une structure MQMDE et l'insère *entre* la structure MQDLH et les données qui la suivent. Dans le MQMD conservé par le gestionnaire de files d'attente avec le message, les zones version-2 sont définies sur leurs valeurs par défaut.

Plusieurs des zones qui existent dans le MQMD version-2 mais pas dans le MQMD version-1 sont des zones d'entrée-sortie dans MQPUT et MQPUT1. Toutefois, le gestionnaire de files d'attente ne renvoie aucune valeur dans les zones équivalentes de MQMDE dans la sortie des appels MQPUT et MQPUT1 ; si l'application requiert ces valeurs de sortie, elle doit utiliser un MQMD version-2 .

## MQMDE renvoyé par l'appel MQGET

Dans l'appel MQGET, si l'application fournit un MQMD version-1 , le gestionnaire de files d'attente préfixe le message renvoyé avec un MQMDE, mais uniquement si une ou plusieurs des zones du MQMDE ont une valeur autre que la valeur par défaut. Le gestionnaire de files d'attente définit la zone *Format* dans MQMD sur la valeur MQFMT\_MD\_EXTENSION pour indiquer qu'un MQMDE est présent.

Si l'application fournit un MQMDE au début du paramètre **Buffer** , MQMDE est ignoré. En retour de l'appel MQGET, il est remplacé par le MQMDE pour le message (si nécessaire) ou remplacé par les données de message d'application (si le MQMDE n'est pas nécessaire).

Si l'appel MQGET renvoie un MQMDE, les données du MQMDE se trouvent généralement dans le jeu de caractères et le codage du gestionnaire de files d'attente. Toutefois, le MQMDE peut être dans un autre jeu de caractères et codage si:

- Le MQMDE a été traité comme des données sur l'appel MQPUT ou MQPUT1 (voir [Tableau 502](#), à la page 493 pour connaître les circonstances qui peuvent en être la cause).
- Le message a été reçu d'un gestionnaire de files d'attente éloignées connecté par une connexion TCP et l'agent MCA (Message Channel Agent) récepteur n'a pas été configuré correctement.

**Remarque :** Sous Windows, les applications compilées avec Micro Focus COBOL utilisent une valeur de MQENC\_NATIVE différente du codage du gestionnaire de files d'attente (voir ci-dessus).

## MQMDE dans les messages des files d'attente de transmission

Les messages des files d'attente de transmission sont préfixés avec la structure MQXQH, qui contient un MQMD version-1 . Un MQMDE peut également être présent, positionné entre la structure MQXQH et les données de message d'application, mais il n'est généralement présent que si une ou plusieurs des zones du MQMDE ont une valeur autre que la valeur par défaut.

D'autres structures d'en-tête MQ peuvent également se produire entre la structure MQXQH et les données de message d'application. Par exemple, lorsque l'en-tête de rebut MQDLH est présent et que le message n'est pas un segment, l'ordre est le suivant:

- MQXQH (contenant un MQMD version-1 )
- MQMDE
- MQDLH
- données de message d'application

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Tableau 503. Zones de MQMDE pour MQMDE		
Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID_STRUCDE_MQM	'MDE↵'
<u>Version</u> (numéro de version de la structure)	MQMDE_VERSION_2	2
<u>StrucLength</u> (longueur de la structure MQMDE)	MQMDE_LENGTH_2	72

Tableau 503. Zones de MQMDE pour MQMDE (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
Codage (codage numérique des données qui suit MQMDE)	MQENC_NATIVE	Dépend de l'environnement
CodedCharSetId (identificateur de jeu de caractères des données qui suivent MQMDE)	MQCCSI_UNDEFINI	0
Format (nom de format des données qui suivent MQMDE)	MQFMT_AUCUN	Espaces vides
Indicateurs (indicateurs généraux)	MQMDEF_AUCUN	0
GroupId (identificateur de groupe)	MQGI_AUCUN	Valeurs NULL
MsgSeqNuméro (numéro de séquence du message logique au sein du groupe)	Aucun	1
Décalage (décalage des données dans le message physique à partir du début du message logique)	Aucun	0
MsgFlags (indicateurs de message)	MQMF_AUCUN	0
OriginalLength (longueur du message d'origine)	MQOL_UNDEFINI	-1
<p><b>Remarques :</b></p> <ol style="list-style-type: none"> <li>1. Le symbole ↵ représente un caractère blanc unique.</li> <li>2. Dans le langage de programmation C, la variable macroMQMDE_DEFAULT contient les valeurs répertoriées dans la table. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</li> </ol> <pre style="background-color: #f0f0f0; padding: 10px;">MQMDE MyMDE = {MQMDE_DEFAULT};</pre>		

## Déclarations de langue

### Déclaration C pour MQMDE

```
typedef struct tagMQMDE MQMDE;
struct tagMQMDE {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    StrucLength;      /* Length of MQMDE structure */
    MQLONG    Encoding;        /* Numeric encoding of data that follows
                               MQMDE */
    MQLONG    CodedCharSetId;  /* Character-set identifier of data that
                               follows MQMDE */
    MQCHAR8   Format;           /* Format name of data that follows
                               MQMDE */
    MQLONG    Flags;            /* General flags */
    MQBYTE24  GroupId;         /* Group identifier */
    MQLONG    MsgSeqNumber;    /* Sequence number of logical message
                               within group */
    MQLONG    Offset;           /* Offset of data in physical message from
                               start of logical message */
    MQLONG    MsgFlags;        /* Message flags */
    MQLONG    OriginalLength;  /* Length of original message */
};
```

## Déclaration COBOL pour MQMDE

```
** MQMDE structure
10 MQMDE.
** Structure identifier
15 MQMDE-STRUCID PIC X(4).
** Structure version number
15 MQMDE-VERSION PIC S9(9) BINARY.
** Length of MQMDE structure
15 MQMDE-STRUCLength PIC S9(9) BINARY.
** Numeric encoding of data that follows MQMDE
15 MQMDE-ENCODING PIC S9(9) BINARY.
** Character-set identifier of data that follows MQMDE
15 MQMDE-CODEDCHARSETID PIC S9(9) BINARY.
** Format name of data that follows MQMDE
15 MQMDE-FORMAT PIC X(8).
** General flags
15 MQMDE-FLAGS PIC S9(9) BINARY.
** Group identifier
15 MQMDE-GROUPID PIC X(24).
** Sequence number of logical message within group
15 MQMDE-MSGSEQNUMBER PIC S9(9) BINARY.
** Offset of data in physical message from start of logical message
15 MQMDE-OFFSET PIC S9(9) BINARY.
** Message flags
15 MQMDE-MSGFLAGS PIC S9(9) BINARY.
** Length of original message
15 MQMDE-ORIGINALLENGTH PIC S9(9) BINARY.
```

## Déclaration PL/I pour MQMDE

```
dcl
1 MQMDE based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 StrucLength fixed bin(31), /* Length of MQMDE structure */
3 Encoding fixed bin(31), /* Numeric encoding of data that
follows MQMDE */
3 CodedCharSetId fixed bin(31), /* Character-set identifier of data
that follows MQMDE */
3 Format char(8), /* Format name of data that follows
MQMDE */
3 Flags fixed bin(31), /* General flags */
3 GroupId char(24), /* Group identifier */
3 MsgSeqNumber fixed bin(31), /* Sequence number of logical message
within group */
3 Offset fixed bin(31), /* Offset of data in physical message
from start of logical message */
3 MsgFlags fixed bin(31), /* Message flags */
3 OriginalLength fixed bin(31); /* Length of original message */
```

## Déclaration High Level Assembler pour MQMDE

```
MQMDE DSECT
MQMDE_STRUCID DS CL4 Structure identifier
MQMDE_VERSION DS F Structure version number
MQMDE_STRUCLength DS F Length of MQMDE structure
MQMDE_ENCODING DS F Numeric encoding of data that follows
* MQMDE
MQMDE_CODEDCHARSETID DS F Character-set identifier of data that
* follows MQMDE
MQMDE_FORMAT DS CL8 Format name of data that follows MQMDE
MQMDE_FLAGS DS F General flags
MQMDE_GROUPID DS XL24 Group identifier
MQMDE_MSGSEQNUMBER DS F Sequence number of logical message
* within group
MQMDE_OFFSET DS F Offset of data in physical message from
* start of logical message
MQMDE_MSGFLAGS DS F Message flags
MQMDE_ORIGINALLENGTH DS F Length of original message
*
MQMDE_LENGTH EQU *-MQMDE
ORG MQMDE
MQMDE_AREA DS CL(MQMDE_LENGTH)
```



## Déclaration Visual Basic pour MQMDE

```
Type MQMDE
  StrucId      As String*4 'Structure identifier'
  Version     As Long      'Structure version number'
  StrucLength As Long      'Length of MQMDE structure'
  Encoding    As Long      'Numeric encoding of data that follows'
                    'MQMDE'
  CodedCharSetId As Long    'Character-set identifier of data that'
                    'follows MQMDE'
  Format       As String*8 'Format name of data that follows MQMDE'
  Flags       As Long      'General flags'
  GroupId     As MQBYTE24 'Group identifier'
  MsgSeqNumber As Long      'Sequence number of logical message within'
                    'group'
  Offset      As Long      'Offset of data in physical message from'
                    'start of logical message'
  MsgFlags    As Long      'Message flags'
  OriginalLength As Long    'Length of original message'
End Type
```

### **StrucId (MQCHAR4) pour MQMDE**

Il s'agit de l'identificateur de structure de la structure d'extension du descripteur de message. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQMDE\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUCDE\_MQM**

Identificateur de la structure d'extension du descripteur de message.

Pour le langage de programmation C, la constante MQMDE\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQMDE\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQMDE**

Il s'agit du numéro de version de la structure ; la valeur doit être:

#### **MQMDE\_VERSION\_2**

Structure d'extension du descripteur de message Version-2 .

La constante suivante indique le numéro de version de la version en cours:

#### **MQMDE\_CURRENT\_VERSION**

Version actuelle de la structure d'extension du descripteur de message.

La valeur initiale de cette zone est MQMDE\_VERSION\_2.

### **StrucLength (MQLONG) pour MQMDE**

Longueur de la structure MQMDE ; la valeur suivante est définie:

#### **MQMDE\_LENGTH\_2**

Longueur de la structure d'extension du descripteur de message version-2 .

La valeur initiale de cette zone est MQMDE\_LENGTH\_2.

### **Codage (MQLONG) pour MQMDE**

Indique le codage numérique des données qui suivent la structure MQMDE ; il ne s'applique pas aux données numériques de la structure MQMDE elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Le gestionnaire de files d'attente ne vérifie pas que la zone est valide. Pour plus d'informations sur les codages de données, voir la zone *Encoding* décrite dans [«MQMD-Descripteur de message»](#), à la page 438 .

La valeur initiale de cette zone est MQENC\_NATIVE.

## ***CodedCharSetId (MQLONG) pour MQMDE***

Indique l'identificateur de jeu de caractères des données qui suivent la structure MQMDE ; il ne s'applique pas aux données de type caractères de la structure MQMDE elle-même.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données. Le gestionnaire de files d'attente ne vérifie pas que cette zone est valide. La valeur spéciale suivante peut être utilisée:

### **MQCCSI\_HÉRITER**

Les données de type caractères dans les données *suivant* cette structure se trouvent dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, la valeur MQCCSI\_INHERIT n'est pas renvoyée par l'appel MQGET.

MQCCSI\_INHERIT ne peut pas être utilisé si la valeur de la zone *PutApplType* dans MQMD est MQAT\_BROKER.

Cette valeur est prise en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour les clients IBM MQ connectés à ces systèmes.

La valeur initiale de cette zone est MQCCSI\_UNDEFINED.

## ***Format (MQCHAR8) pour MQMDE***

Indique le nom de format des données qui suivent la structure MQMDE.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données. Le gestionnaire de files d'attente ne vérifie pas que cette zone est valide. Pour plus d'informations sur les noms de format, voir la zone *Format* décrite dans [«MQMD-Descripteur de message»](#), à la page 438.

La valeur initiale de cette zone est MQFMT\_NONE.

## ***Indicateurs (MQLONG) pour MQMDE***

L'indicateur suivant peut être spécifié:

### **MQMDEF\_AUCUN**

Aucun indicateur.

La valeur initiale de cette zone est MQMDEF\_NONE.

## ***GroupId (MQBYTE24) pour MQMDE***

Voir la zone *GroupId* décrite dans [«MQMD-Descripteur de message»](#), à la page 438. La valeur initiale de cette zone est MQGI\_NONE.

## ***MsgSeqNuméro (MQLONG) pour MQMDE***

Voir la zone *MsgSeqNumber* décrite dans [«MQMD-Descripteur de message»](#), à la page 438. La valeur initiale de cette zone est 1.

## ***Décalage (MQLONG) pour MQMDE***

Voir la zone *Offset* décrite dans «MQMD-Descripteur de message», à la page 438. La valeur initiale de cette zone est 0.

### **MsgFlags (MQLONG) pour MQMDE**

Voir la zone *MsgFlags* décrite dans «MQMD-Descripteur de message», à la page 438. La valeur initiale de cette zone est MQMF\_NONE.

### **OriginalLength (MQLONG) pour MQMDE**

Voir la zone *OriginalLength* décrite dans «MQMD-Descripteur de message», à la page 438. La valeur initiale de cette zone est MQOL\_UNDEFINED.

## **MQMHBO-Options de traitement des messages vers la mémoire tampon**

La structure MQMHBO permet aux applications de spécifier des options qui contrôlent la façon dont les mémoires tampon sont produites à partir des descripteurs de message. La structure est un paramètre d'entrée dans l'appel MQMHBUF.

### **Jeu de caractères et codage**

Les données de MQMHBO doivent être dans le jeu de caractères de l'application et dans le codage de l'application (MQENC\_NATIVE).

### **Zones**

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Tableau 504. Zones dans MQMHBO		
Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
StrucId (identificateur de structure)	MQMHBO_STRUC_ID	'MHBO'
Version (numéro de version de la structure)	MQMHBO_VERSION_1	1
Options (options contrôlant l'action de MQMHBUF)	MQMHBO_PROPERTIES_IN_MQRFH2	
<p><b>Remarques :</b></p> <ol style="list-style-type: none"> <li>1. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.</li> <li>2. Dans le langage de programmation C, la variable macro MQMHBO_DEFAULT contient les valeurs répertoriées dans la table. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</li> </ol> <pre>MQMHBO MyMHBO = {MQMHBO_DEFAULT};</pre>		

## **Déclarations de langue**

Déclaration C pour MQMHBO

```
typedef struct tagMQMHBO MQMHBO;
struct tagMQMHBO {
    MQCHAR4  StrucId;          /* Structure identifier */
    MQLONG   Version;        /* Structure version number */
    MQLONG   Options;        /* Options that control the action of
```

```
};
MQMHBUF */
```

## Déclaration COBOL pour MQMHBO

```
** MQMHBO structure
10 MQMHBO.
**   Structure identifier
15 MQMHBO-STRUCID          PIC X(4).
**   Structure version number
15 MQMHBO-VERSION        PIC S9(9) BINARY.
**   Options that control the action of MQMHBUF
15 MQMHBO-OPTIONS        PIC S9(9) BINARY.
```

## Déclaration PL/I pour MQMHBO

```
Dcl
  1 MQMHBO based,
  3 StrucId      char(4),          /* Structure identifier */
  3 Version      fixed bin(31),   /* Structure version number */
  3 Options      fixed bin(31),   /* Options that control the action
                                of MQMHBUF */
```

## Déclaration High Level Assembler pour MQMHBO

```
MQMHBO          DSECT
MQMHBO_STRUCID  DS   CL4  Structure identifier
MQMHBO_VERSION  DS   F    Structure version number
MQMHBO_OPTIONS  DS   F    Options that control the
*                  action of MQMHBUF
MQMHBO_LENGTH   EQU  *-MQMHBO
MQMHBO_AREA     DS   CL(MQMHBO_LENGTH)
```

### **StrucId (MQCHAR4) pour MQMHBO**

Il s'agit de l'identificateur de structure du descripteur de message pour la structure des options de mémoire tampon. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQMHBO\_STRUC\_ID.

La valeur doit être:

#### **MQMHBO\_STRUC\_ID**

Identificateur de la structure des options de traitement des messages vers la mémoire tampon.

Pour le langage de programmation C, la constante MQMHBO\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQMHBO\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQMHBO**

Descripteur de message pour la structure des options de mémoire tampon-Zone Version

Il s'agit du numéro de version de la structure. La valeur doit être:

#### **MQMHBO\_VERSION\_1**

Numéro de version de la structure des options de traitement des messages à la mémoire tampon.

La constante suivante indique le numéro de version de la version en cours:

#### **MQMHBO\_CURRENT\_VERSION**

Version actuelle de la structure des options du descripteur de message vers la mémoire tampon.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQMHBO\_VERSION\_1.

### **Options (MQLONG) pour MQMHBO**

Descripteur de message pour la structure des options de mémoire tampon-Zone Options

Ces options contrôlent l'action de MQMHBUF.

Vous devez spécifier l'option suivante:

### **MQMHBO\_PROPERTIES\_IN\_MQRFH2**

Lors de la conversion des propriétés d'un descripteur de message en mémoire tampon, convertissez-les au format MQRFH2 .

Si vous le souhaitez, vous pouvez également spécifier l'option suivante. Pour spécifier plusieurs options, ajoutez les valeurs ensemble (n'ajoutez pas la même constante plusieurs fois) ou combinez les valeurs à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations bit).

### **MQMHBO\_DELETE\_PROPERTIES**

Les propriétés ajoutées à la mémoire tampon sont supprimées du descripteur de message. Si l'appel échoue, aucune propriété n'est supprimée.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQMHBO\_PROPERTIES\_IN\_MQRFH2.

## **MQOD-Descripteur d'objet**

La structure MQOD est utilisée pour spécifier un objet par nom. La structure est un paramètre d'entrée-sortie dans les appels MQOPEN et MQPUT1 .

Les types d'objet suivants sont valides:

- File d'attente ou liste de distribution
- Liste de noms
- Définition de processus
- Gestionnaire de files d'attente
- Topic

## **Disponibilité**

Tous les systèmes IBM MQ , plus IBM MQ MQI clients connectés à ces systèmes.

## **Version**

La version actuelle de MQOD est MQOD\_VERSION\_4. Les applications que vous souhaitez porter entre plusieurs environnements doivent s'assurer que la version requise de MQOD est prise en charge dans tous les environnements concernés. Les zones qui existent uniquement dans les versions les plus récentes de la structure sont identifiées comme telles dans les descriptions qui suivent.

Les fichiers d'en-tête, COPY et INCLUDE fournis pour les langages de programmation pris en charge contiennent la version la plus récente de MQOD prise en charge par l'environnement, mais avec la valeur initiale de la zone *Version* définie sur MQOD\_VERSION\_1. Pour utiliser des zones qui ne sont pas présentes dans la structure version-1 , l'application doit définir la zone *Version* sur le numéro de version de la version requise.

Pour ouvrir une liste de distribution, *Version* doit avoir la valeur MQOD\_VERSION\_2 ou une valeur supérieure.

## **Jeu de caractères et codage**

Les données dans MQOD doivent se trouver dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par MQENC\_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

## **Zones**

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID_STRUC_MQODE	'OD--'
<u>Version</u> (numéro de version de la structure)	MQOD_VERSION_1	1
<u>ObjectType</u> (type d'objet)	MQOT_Q	1
<u>ObjectName</u> (nom de l'objet)	Aucun	Chaîne nulle ou blancs
<u>ObjectQMgrNom</u> (nom du gestionnaire de files d'attente d'objet)	Aucun	Chaîne nulle ou blancs
<u>DynamicQName</u> (nom de la file d'attente dynamique)	Aucun	'CSQ.*' sous z/OS ; 'AMQ.*' dans le cas contraire
<u>IDAlternateUser</u> (autre ID utilisateur)	Aucun	Chaîne nulle ou blancs
<b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieur à MQOD_VERSION_2.		
<u>RecsPresent</u> (nombre d'enregistrements d'objet présents)	Aucun	0
<u>KnownDestNombre</u> (nombre de files d'attente locales ouvertes avec succès)	Aucun	0
<u>UnknownDestfiles</u> d'attente éloignées ouvertes (nombre de files d'attente éloignées ouvertes avec succès)	Aucun	0
<u>NombreInvalidDest</u> (nombre de files d'attente dont l'ouverture a échoué)	Aucun	0
<u>ObjectRecDécalage</u> (décalage du premier enregistrement d'objet à partir du début de MQOD)	Aucun	0
<u>ResponseRecde</u> de réponse (décalage du premier enregistrement de réponse à partir du début de MQOD)	Aucun	0
<u>ObjectRecPtr</u> (adresse du premier enregistrement d'objet)	Aucun	Pointeur null ou octets null
<u>ResponseRecPtr</u> (adresse du premier enregistrement de réponse)	Aucun	Pointeur null ou octets null
<b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieur à MQOD_VERSION_3.		
<u>IDAlternateSecurity</u> (autre identificateur de sécurité)	MQSID_AUCUN	Valeurs NULL
<u>ResolvedQName</u> (nom de la file d'attente résolue)	Aucun	Chaîne nulle ou blancs
<u>ResolvedQMgrNom</u> (nom du gestionnaire de files d'attente résolu)	Aucun	Chaîne nulle ou blancs
<b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieur à MQOD_VERSION_4.		
<u>ObjectString</u> (nom d'objet long)	MQCHARV_VALEUR PAR DEFAUT	Comme défini pour MQCHARV

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>SelectionString</u> (chaîne de sélection)	MQCHARV_VALEUR PAR DEFAULT	Comme défini pour MQCHARV
<u>ResObjectString</u> (nom d'objet long résolu)	MQCHARV_VALEUR PAR DEFAULT	Comme défini pour MQCHARV
<u>ResolvedType</u> (type d'objet résolu)	MQOT_AUCUN	0
<p><b>Remarques :</b></p> <ol style="list-style-type: none"> <li>1. Le symbole ~ représente un caractère blanc unique.</li> <li>2. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.</li> <li>3. Dans le langage de programmation C, la variable macroMQOD_DEFAULT contient les valeurs répertoriées dans la table. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</li> </ol> <pre style="background-color: #f0f0f0; padding: 10px;">MQOD MyOD = {MQOD_DEFAULT};</pre>		

## Déclarations de langue

### Déclaration C pour MQOD

```
typedef struct tagMQOD MQOD;
struct tagMQOD {
    MQCHAR4      StrucId;           /* Structure identifier */
    MQLONG       Version;          /* Structure version number */
    MQLONG       ObjectType;       /* Object type */
    MQCHAR48     ObjectName;       /* Object name */
    MQCHAR48     ObjectQMgrName;   /* Object queue manager name */
    MQCHAR48     DynamicQName;    /* Dynamic queue name */
    MQCHAR12     AlternateUserId;  /* Alternate user identifier */
    /* Ver:1 */
    MQLONG       RecsPresent;      /* Number of object records present */
    MQLONG       KnownDestCount;  /* Number of local queues opened
    successfully */
    MQLONG       UnknownDestCount; /* Number of remote queues opened
    successfully */
    MQLONG       InvalidDestCount; /* Number of queues that failed to
    open */
    MQLONG       ObjectRecOffset; /* Offset of first object record from
    start of MQOD */
    MQLONG       ResponseRecOffset; /* Offset of first response record
    from start of MQOD */
    MQPTR        ObjectRecPtr;     /* Address of first object record */
    MQPTR        ResponseRecPtr;  /* Address of first response record */
    /* Ver:2 */
    MQBYTE40     AlternateSecurityId; /* Alternate security identifier */
    MQCHAR48     ResolvedQName;     /* Resolved queue name */
    MQCHAR48     ResolvedQMgrName; /* Resolved queue manager name */
    /* Ver:3 */
    MQCHARV      ObjectString;     /* Object Long name */
    MQCHARV      SelectionString;  /* Message Selector */
    MQCHARV      ResObjectString;  /* Resolved Long object name*/
    MQLONG       ResolvedType      /* Alias queue resolved
    object type */
    /* Ver:4 */
};
```

### Déclaration COBOL pour MQOD

```
** MQOD structure
10 MQOD.
```

```

** Structure identifier
 15 MQOD-STRUCID PIC X(4).
** Structure version number
 15 MQOD-VERSION PIC S9(9) BINARY.
** Object type
 15 MQOD-OBJECTTYPE PIC S9(9) BINARY.
** Object name
 15 MQOD-OBJECTNAME PIC X(48).
** Object queue manager name
 15 MQOD-OBJECTQMGRNAME PIC X(48).
** Dynamic queue name
 15 MQOD-DYNAMICQNAME PIC X(48).
** Alternate user identifier
 15 MQOD-ALTERNATEUSERID PIC X(12).
** Number of object records present
 15 MQOD-RECSPRESENT PIC S9(9) BINARY.
** Number of local queues opened successfully
 15 MQOD-KNOWNDDESTCOUNT PIC S9(9) BINARY.
** Number of remote queues opened successfully
 15 MQOD-UNKNOWNDDESTCOUNT PIC S9(9) BINARY.
** Number of queues that failed to open
 15 MQOD-INVALIDDESTCOUNT PIC S9(9) BINARY.
** Offset of first object record from start of MQOD
 15 MQOD-OBJECTRECOFFSET PIC S9(9) BINARY.
** Offset of first response record from start of MQOD
 15 MQOD-RESPONSERECOFFSET PIC S9(9) BINARY.
** Address of first object record
 15 MQOD-OBJECTRECPTN POINTER.
** Address of first response record
 15 MQOD-RESPONSERECPTN POINTER.
** Alternate security identifier
 15 MQOD-ALTERNATESECURITYID PIC X(40).
** Resolved queue name
 15 MQOD-RESOLVEDQNAME PIC X(48).
** Resolved queue manager name
 15 MQOD-RESOLVEDQMGRNAME PIC X(48).
** Object Long name
 15 MQOD-OBJECTSTRING.
** Address of variable length string
 20 MQOD-OBJECTSTRING-VSPTR POINTER.
** Offset of variable length string
 20 MQOD-OBJECTSTRING-VSOFFSET PIC S9(9) BINARY.
** size of buffer
 20 MQOD-OBJECTSTRING-VSBUFSIZE PIC S9(9) BINARY.
** Length of variable length string
 20 MQOD-OBJECTSTRING-VSLENGTH PIC S9(9) BINARY.
** CCSID of variable length string
 20 MQOD-OBJECTSTRING-VSCCSID PIC S9(9) BINARY.
** Message Selector
 15 MQOD-SELECTIONSTRING.
** Address of variable length string
 20 MQOD-SELECTIONSTRING-VSPTR POINTER.
** Offset of variable length string
 20 MQOD-SELECTIONSTRING-VSOFFSET PIC S9(9) BINARY.
** size of buffer
 20 MQOD-SELECTIONSTRING-VSBUFSIZE PIC S9(9) BINARY.
** Length of variable length string
 20 MQOD-SELECTIONSTRING-VSLENGTH PIC S9(9) BINARY.
** CCSID of variable length string
 20 MQOD-SELECTIONSTRING-VSCCSID PIC S9(9) BINARY.
** Resolved Long object name
 15 MQOD-RESOBJECTSTRING.
** Address of variable length string
 20 MQOD-RESOBJECTSTRING-VSPTR POINTER.
** Offset of variable length string
 20 MQOD-RESOBJECTSTRING-VSOFFSET PIC S9(9) BINARY.
** size of buffer
 20 MQOD-RESOBJECTSTRING-VSBUFSIZE PIC S9(9) BINARY.
** Length of variable length string
 20 MQOD-RESOBJECTSTRING-VSLENGTH PIC S9(9) BINARY.
** CCSID of variable length string
 20 MQOD-RESOBJECTSTRING-VSCCSID PIC S9(9) BINARY.
** Alias queue resolved object type
 15 MQOD-RESOLVEDTYPE PIC S9(9) BINARY.

```

## Déclaration PL/I pour MQOD

```

dcl
 1 MQOD based,

```



```

3 StrucId          char(4),          /* Structure identifier */
3 Version         fixed bin(31), /* Structure version number */
3 ObjectType      fixed bin(31), /* Object type */
3 ObjectName     char(48),      /* Object name */
3 ObjectQMgrName char(48),      /* Object queue manager name */
3 DynamicQName   char(48),      /* Dynamic queue name */
3 AlternateUserId char(12),     /* Alternate user identifier */
3 RecsPresent    fixed bin(31), /* Number of object records
present */
3 KnownDestCount fixed bin(31), /* Number of local queues opened
successfully */
3 UnknownDestCount fixed bin(31), /* Number of remote queues opened
successfully */
3 InvalidDestCount fixed bin(31), /* Number of queues that failed to
open */
3 ObjectRecOffset fixed bin(31), /* Offset of first object record
from start of MQOD */
3 ResponseRecOffset fixed bin(31), /* Offset of first response record
from start of MQOD */
3 ObjectRecPtr    pointer,      /* Address of first object record */
3 ResponseRecPtr  pointer,      /* Address of first response
record */
3 AlternateSecurityId char(40), /* Alternate security identifier */
3 ResolvedQName   char(48),      /* Resolved queue name */
3 ResolvedQMgrName char(48), /* Resolved queue manager name */
3 ObjectString,   /* Object Long name */
5 VSPtr          pointer,      /* Address of variable length string */
5 VSOFFSET      fixed bin(31), /* Offset of variable length string */
5 VSBUF_SIZE    fixed bin(31), /* size of buffer */
5 VSLength      fixed bin(31), /* Length of variable length string */
5 VSCCSID       fixed bin(31), /* CCSID of variable length string */
3 SelectionString, /* Message Selection */
5 VSPtr          pointer,      /* Address of variable length string */
5 VSOFFSET      fixed bin(31), /* Offset of variable length string */
5 VSBUF_SIZE    fixed bin(31), /* size of buffer */
5 VSLength      fixed bin(31), /* Length of variable length string */
5 VSCCSID       fixed bin(31), /* CCSID of variable length string */
3 ResObjectString, /* Resolved Long object name */
5 VSPtr          pointer,      /* Address of variable length string */
5 VSOFFSET      fixed bin(31), /* Offset of variable length string */
5 VSBUF_SIZE    fixed bin(31), /* size of buffer */
5 VSLength      fixed bin(31), /* Length of variable length string */
5 VSCCSID       fixed bin(31), /* CCSID of variable length string */
3 ResolvedType   fixed bin(31); /* Alias queue resolved object type */

```

## Déclaration High Level Assembler pour MQOD

```

MQOD              DSECT
MQOD_STRUCID     DS   CL4   Structure identifier
MQOD_VERSION     DS   F     Structure version number
MQOD_OBJECTTYPE  DS   F     Object type
MQOD_OBJECTNAME  DS   CL48  Object name
MQOD_OBJECTQMGRNAME DS   CL48  Object queue manager name
MQOD_DYNAMICQNAME DS   CL48  Dynamic queue name
MQOD_ALTERNATEUSERID DS   CL12  Alternate user identifier
MQOD_RECPRESENT  DS   F     Number of object records present
MQOD_KNOWNDESTCOUNT DS   F     Number of local queues opened
*
MQOD_UNKNOWNDSTCOUNT DS   F     Number of remote queues opened
*
MQOD_INVALIDDESTCOUNT DS   F     Number of queues that failed to
*
MQOD_OBJECTRECOFFSET DS   F     Offset of first object record from
*
MQOD_RESPONSERECOFFSET DS   F     Offset of first response record
*
MQOD_OBJECTRECPTTR DS   F     Address of first object record
MQOD_RESPONSERECPTTR DS   F     Address of first response record
MQOD_ALTERNATESECURITYID DS   XL40  Alternate security identifier
MQOD_RESOLVEDQNAME DS   CL48  Resolved queue name
MQOD_RESOLVEDQMGRNAME DS   CL48  Resolved queue manager name
MQOD_OBJECTSTRING DS   F     Object Long name
MQOD_OBJECTSTRING_VSPTR DS   F     Address of variable length string
MQOD_OBJECTSTRING_VSOFFSET DS   F     Offset of variable length string
MQOD_OBJECTSTRING_VSBUF_SIZE DS   F     size of buffer
MQOD_OBJECTSTRING_VSLength DS   F     Length of variable length string
MQOD_OBJECTSTRING_VSCCSID DS   F     CCSID of variable length string
MQOD_OBJECTSTRING_LENGTH EQU  *- MQOD_OBJECTSTRING
ORG MQOD_OBJECTSTRING

```

```

MQOD_OBJECTSTRING_AREA      DS    CL(MQOD_OBJECTSTRING_LENGTH)
*
MQOD_SELECTIONSTRING        DS    F      Message Selector
MQOD_SELECTIONSTRING_VSPTR  DS    F      Address of variable length string
MQOD_SELECTIONSTRING_VSOFFSET DS    F      Offset of variable length string
MQOD_SELECTIONSTRING_VSBUFFSIZE DS    F      size of buffer
MQOD_SELECTIONSTRING_VSLENGTH DS    F      Length of variable length string
MQOD_SELECTIONSTRING_VSCCSID DS    F      CCSID of variable length string
MQOD_SELECTIONSTRING_LENGTH EQU    *- MQOD_SELECTIONSTRING
MQOD_SELECTIONSTRING_AREA   DS    CL(MQOD_SELECTIONSTRING_LENGTH)
*
MQOD_RESOBJECTSTRING        DS    F      Resolved Long object name
MQOD_RESOBJECTSTRING_VSPTR  DS    F      Address of variable length string
MQOD_RESOBJECTSTRING_VSOFFSET DS    F      Offset of variable length string
MQOD_RESOBJECTSTRING_VSBUFFSIZE DS    F      size of buffer
MQOD_RESOBJECTSTRING_VSLENGTH DS    F      Length of variable length string
MQOD_RESOBJECTSTRING_VSCCSID DS    F      CCSID of variable length string
MQOD_RESOBJECTSTRING_LENGTH EQU    *- MQOD_RESOBJECTSTRING
MQOD_RESOBJECTSTRING_AREA   DS    CL(MQOD_RESOBJECTSTRING_LENGTH)
MQOD_RESOLVEDTYPE           DS    F      Alias queue object resolved type
*
MQOD_LENGTH                 EQU    *- MQOD
                                ORG    MQOD
MQOD_AREA                   DS    CL(MQOD_LENGTH)

```

## Déclaration Visual Basic pour MQOD

```

Type MQOD
  StrucId      As String*4  'Structure identifier'
  Version      As Long      'Structure version number'
  ObjectType    As Long      'Object type'
  ObjectName   As String*48 'Object name'
  ObjectQMgrName As String*48 'Object queue manager name'
  DynamicQName As String*48 'Dynamic queue name'
  AlternateUserId As String*12 'Alternate user identifier'
  RecsPresent   As Long      'Number of object records present'
  KnownDestCount As Long      'Number of local queues opened'
                                'successfully'
  UnknownDestCount As Long      'Number of remote queues opened'
                                'successfully'
  InvalidDestCount As Long      'Number of queues that failed to'
                                'open'
  ObjectRecOffset As Long      'Offset of first object record from'
                                'start of MQOD'
  ResponseRecOffset As Long      'Offset of first response record'
                                'from start of MQOD'
  ObjectRecPtr   As MQPTR      'Address of first object record'
  ResponseRecPtr As MQPTR      'Address of first response record'
  AlternateSecurityId As MBYTE40 'Alternate security identifier'
  ResolvedQName  As String*48 'Resolved queue name'
  ResolvedQMgrName As String*48 'Resolved queue manager name'
End Type

```

### **StrucId (MQCHAR4) pour MQOD**

Il s'agit de l'identificateur de structure de la structure de descripteur d'objet. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQOD\_STRUC\_ID.

La valeur doit être:

### **ID\_STRUC\_MQODE**

Identificateur de la structure de descripteur d'objet.

Pour le langage de programmation C, la constante MQOD\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQOD\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQOD**

Il s'agit du numéro de version de la structure ; la valeur doit être l'une des suivantes:

### **MQOD\_VERSION\_1**

Structure de descripteur d'objet Version-1 .

**MQOD\_VERSION\_2**

Structure de descripteur d'objet Version-2 .

**MQOD\_VERSION\_3**

Structure de descripteur d'objet Version-3 .

**MQOD\_VERSION\_4**

Structure de descripteur d'objet Version-4 .

Toutes les versions sont prises en charge dans tous les environnements IBM MQ V7.0 .

Les zones qui existent uniquement dans les versions plus récentes de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

**VERSION MQOD\_CURRENT\_VERSION**

Version actuelle de la structure de descripteur d'objet.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQOD\_VERSION\_1.

**ObjectType (MQLONG) pour MQOD**

Type d'objet nommé dans le descripteur d'objet. Les valeurs possibles sont les suivantes :

**MQOT\_CLNTCONN\_CHANNEL**

Canal de connexion client. Le nom de l'objet se trouve dans la zone *ObjectName* .

**MQOT\_Q**

File d'attente. Le nom de l'objet se trouve dans la zone *ObjectName* .

**MQOT\_NAMELIST**

NAMELIST. Le nom de l'objet se trouve dans la zone *ObjectName*

**PROCESSUS MQ**

Définition de processus. Le nom de l'objet se trouve dans la zone *ObjectName*

**MQOT\_Q\_DIR**

Gestionnaire de files d'attente. Le nom de l'objet se trouve dans la zone *ObjectName*

**MQOT\_TOPIC**

:NONE. Le nom complet de la rubrique peut être généré à partir de deux zones différentes: *ObjectName* et *ObjectString*.

Pour plus de détails sur l'utilisation de ces deux zones, voir [Combinaison de chaînes de rubrique](#).

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQOT\_Q.

**ObjectName (MQCHAR48) pour MQOD**

Il s'agit du nom local de l'objet tel qu'il est défini sur le gestionnaire de files d'attente identifié par *ObjectQMgrName*. Le nom peut contenir les caractères suivants:

- Caractères alphabétiques en majuscules (A à Z)
- Caractères alphabétiques minuscules (a à z)
- Chiffres (0 à 9)
- Point (.), barre oblique (/), trait de soulignement (\_), pourcentage (%)

Le nom ne doit pas contenir d'espaces de début ou d'espaces imbriqués, mais peut contenir des espaces de fin. Utilisez un caractère null pour indiquer la fin des données significatives dans le nom ; la valeur null et les caractères qui la suivent sont traités comme des blancs. Les restrictions suivantes s'appliquent dans les environnements indiqués:

- Sur les systèmes qui utilisent EBCDIC Katakana, les caractères minuscules ne peuvent pas être utilisés.
- Sous z/OS :
  - Évitez les noms qui commencent ou se terminent par un trait de soulignement ; ils ne peuvent pas être traités par les panneaux d'opérations et de contrôle.

- Le caractère de pourcentage a une signification spéciale pour RACF. Si RACF est utilisé comme gestionnaire de sécurité externe, les noms ne doivent pas contenir le pourcentage. Si tel est le cas, ces noms ne sont pas inclus dans les contrôles de sécurité lorsque des profils génériques RACF sont utilisés.
- Sous IBM i, les noms contenant des caractères minuscules, des barres obliques ou des pourcentages doivent être placés entre guillemets lorsqu'ils sont spécifiés dans les commandes. Ces guillemets ne doivent pas être indiqués pour les noms qui apparaissent en tant que zones dans les structures ou en tant que paramètres dans les appels.

Le nom complet de la rubrique peut être généré à partir de deux zones différentes: *ObjectName* et *ObjectString*. Pour plus de détails sur l'utilisation de ces deux zones, voir [Combinaison de chaînes de rubrique](#).

Les points suivants s'appliquent aux types d'objet indiqués:

- Si *ObjectName* est le nom d'une file d'attente modèle, le gestionnaire de files d'attente crée une file d'attente dynamique avec les attributs de la file d'attente modèle et renvoie dans la zone *ObjectName* le nom de la file d'attente créée. Une file d'attente modèle ne peut être indiquée que sur l'appel MQOPEN ; une file d'attente modèle n'est pas valide sur l'appel MQPUT1 .
- Si *ObjectName* est le nom d'une file d'attente alias avec TARGTYPE (TOPIC), un contrôle de sécurité est d'abord effectué sur la file d'attente alias nommée ; cela est normal lorsque des files d'attente alias sont utilisées. Lorsque le contrôle de sécurité aboutit, l'appel MQOPEN se poursuit et se comporte comme un appel MQOPEN sur un MQOT\_TOPIC; cela inclut un contrôle de sécurité par rapport à l'objet de rubrique d'administration.
- Si *ObjectName* et *ObjectQMgrName* identifient une file d'attente partagée appartenant au groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local, il ne doit pas y avoir de définition de file d'attente du même nom sur le gestionnaire de files d'attente local. S'il existe une telle définition (file d'attente locale, file d'attente alias, file d'attente éloignée ou file d'attente modèle), l'appel échoue avec le code anomalie MQRC\_OBJECT\_NOT\_UNIQUE.
- Si l'objet en cours d'ouverture est une liste de distribution (c'est-à-dire si *RecsPresent* est présent et supérieur à zéro), *ObjectName* doit être vide ou la chaîne null. Si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie MQRC\_OBJECT\_NAME\_ERROR.
- Si *ObjectType* est MQOT\_Q\_MGR, des règles spéciales s'appliquent ; dans ce cas, le nom doit être entièrement vide jusqu'au premier caractère null ou jusqu'à la fin de la zone.

Il s'agit d'une zone d'entrée-sortie pour l'appel MQOPEN lorsque *ObjectName* est le nom d'une file d'attente modèle et d'une zone d'entrée uniquement dans tous les autres cas. La longueur de cette zone est indiquée par MQ\_Q\_NAME\_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

### **ObjectQMgrNom (MQCHAR48) pour MQOD**

Il s'agit du nom du gestionnaire de files d'attente sur lequel l'objet *ObjectName* est défini. Les caractères valides dans le nom sont identiques à ceux de *ObjectName* (voir «[ObjectName \(MQCHAR48\) pour MQOD](#)», à la page 507 ). Un nom entièrement vide jusqu'au premier caractère NULL ou jusqu'à la fin de la zone indique le gestionnaire de files d'attente auquel l'application est connectée (le gestionnaire de files d'attente local).

Les points suivants s'appliquent aux types d'objet indiqués:

- Si *ObjectType* est MQOT\_TOPIC, MQOT\_NAMELIST, MQOT\_PROCESS ou MQOT\_Q\_MGR, *ObjectQMgrName* doit être vide ou le nom du gestionnaire de files d'attente local.
- Si *ObjectName* est le nom d'une file d'attente modèle, le gestionnaire de files d'attente crée une file d'attente dynamique avec les attributs de la file d'attente modèle et renvoie dans la zone *ObjectQMgrName* le nom du gestionnaire de files d'attente sur lequel la file d'attente est créée ; il s'agit du nom du gestionnaire de files d'attente local. Une file d'attente modèle ne peut être indiquée que sur l'appel MQOPEN ; une file d'attente modèle n'est pas valide sur l'appel MQPUT1 .
- Si *ObjectName* est le nom d'une file d'attente de cluster et que *ObjectQMgrName* est vide, la destination des messages envoyés à l'aide de l'identificateur de file d'attente renvoyé par l'appel

MQOPEN est choisie par le gestionnaire de files d'attente (ou l'exit de charge de travail de cluster, s'il est installé) comme suit:

- Si MQOO\_BIND\_ON\_OPEN est spécifié, le gestionnaire de files d'attente sélectionne une instance particulière de la file d'attente de cluster lors du traitement de l'appel MQOPEN et tous les messages insérés à l'aide de cet identificateur de file d'attente sont envoyés à cette instance.
- Si MQOO\_BIND\_NOT\_FIXED est spécifié, le gestionnaire de files d'attente peut choisir une autre instance de la file d'attente de destination (résidant sur un gestionnaire de files d'attente différent dans le cluster) pour chaque appel MQPUT successif qui utilise ce descripteur de file d'attente.

Si l'application doit envoyer un message à une instance *spécifique* d'une file d'attente de cluster (c'est-à-dire une instance de file d'attente résidant sur un gestionnaire de files d'attente particulier du cluster), l'application doit spécifier le nom de ce gestionnaire de files d'attente dans la zone *ObjectQMgrName*. Cela force le gestionnaire de files d'attente local à envoyer le message au gestionnaire de files d'attente de destination spécifié.

- Si *ObjectName* est le nom d'une file d'attente partagée qui est détenu par le groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local, *ObjectQMgrName* peut être le nom du groupe de partage de files d'attente, le nom du gestionnaire de files d'attente local ou vide ; le message est placé dans la même file d'attente, selon la valeur spécifiée.

Les groupes de partage de files d'attente sont pris en charge uniquement sous z/OS.

- Si *ObjectName* est le nom d'une file d'attente partagée appartenant à un groupe de partage de files d'attente éloignées (c'est-à-dire un groupe de partage de files d'attente auquel le gestionnaire de files d'attente local n'appartient pas), *ObjectQMgrName* doit être le nom du groupe de partage de files d'attente. Vous pouvez utiliser le nom d'un gestionnaire de files d'attente appartenant à ce groupe, mais cela peut retarder le message si ce gestionnaire de files d'attente particulier n'est pas disponible lorsque le message arrive au groupe de partage de files d'attente.
- Si l'objet ouvert est une liste de distribution (c'est-à-dire que *RecsPresent* est supérieur à zéro), *ObjectQMgrName* doit être vide ou la chaîne null. Si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie MQRC\_OBJECT\_Q\_MGR\_NAME\_ERROR.

Il s'agit d'une zone d'entrée-sortie pour l'appel MQOPEN lorsque *ObjectName* est le nom d'une file d'attente modèle et d'une zone d'entrée uniquement dans tous les autres cas. La longueur de cette zone est indiquée par MQ\_Q\_MGR\_NAME\_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

### **DynamicQName (MQCHAR48) pour MQOD**

Il s'agit du nom d'une file d'attente dynamique qui doit être créée par l'appel MQOPEN. Cela n'est pertinent que lorsque *ObjectName* spécifie le nom d'une file d'attente modèle ; dans tous les autres cas, *DynamicQName* est ignoré.

Les caractères valides dans le nom sont identiques à ceux de *ObjectName*, sauf qu'un astérisque est également valide. Un nom à blanc (ou dans lequel seuls des blancs apparaissent avant le premier caractère nul) n'est pas valide si *ObjectName* est le nom d'une file d'attente modèle.

Si le dernier caractère non blanc du nom est un astérisque (\*), le gestionnaire de files d'attente le remplace par une chaîne de caractères qui garantit que le nom généré pour la file d'attente est unique au niveau du gestionnaire de files d'attente local. Pour autoriser un nombre suffisant de caractères, l'astérisque est admis uniquement aux positions 1 à 33. Il ne doit pas y avoir d'autres caractères que des blancs ou un caractère nul après l'astérisque.

L'astérisque peut apparaître à la première position de caractère, auquel cas le nom se compose uniquement des caractères générés par le gestionnaire de files d'attente.

Sous z/OS, n'utilisez pas d'astérisque en première position, car aucun contrôle de sécurité ne peut être effectué sur une file d'attente dont le nom complet est généré automatiquement.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par MQ\_Q\_NAME\_LENGTH. La valeur initiale de cette zone est déterminée par l'environnement:

- Sous z/OS, la valeur est 'CSQ.\*'.

- Sur les autres plateformes, la valeur est 'AMQ.\*'.

La valeur est une chaîne à terminaison nulle en C et une chaîne à remplissage vide dans d'autres langages de programmation.

### **ID AlternateUser(MQCHAR12) pour MQOD**

Si vous spécifiez MQOO\_ALTERNATE\_USER\_AUTHORITY pour l'appel MQOPEN ou MQPMO\_ALTERNATE\_USER\_AUTHORITY pour l'appel MQPUT1, cette zone contient un autre identificateur utilisateur qui est utilisé pour vérifier l'autorisation d'ouverture, à la place de l'identificateur utilisateur sous lequel l'application est en cours d'exécution. Cependant, certaines vérifications sont toujours effectuées avec l'identificateur d'utilisateur en cours (par exemple, des vérifications de contexte).

Si MQOO\_ALTERNATE\_USER\_AUTHORITY ou MQPMO\_ALTERNATE\_USER\_AUTHORITY est spécifié et que cette zone est entièrement vide jusqu'au premier caractère null ou à la fin de la zone, l'ouverture ne peut aboutir que si aucune autorisation utilisateur n'est requise pour ouvrir cet objet avec les options spécifiées.

Si ni MQOO\_ALTERNATE\_USER\_AUTHORITY ni MQPMO\_ALTERNATE\_USER\_AUTHORITY n'est spécifié, cette zone est ignorée.

Les différences suivantes existent dans les environnements indiqués:

- Sous z/OS, seuls les 8 premiers caractères de *AlternateUserId* sont utilisés pour vérifier l'autorisation d'ouverture. Toutefois, l'ID utilisateur en cours doit être autorisé à indiquer cet ID utilisateur de remplacement particulier ; les 12 caractères de l'ID utilisateur de remplacement sont utilisés pour cette vérification. L'ID utilisateur ne doit contenir que des caractères autorisés par le gestionnaire de sécurité externe.

Si *AlternateUserId* est spécifié pour une file d'attente, la valeur peut être utilisée ultérieurement par le gestionnaire de files d'attente lors de l'insertion de messages. Si les options MQPMO\_\*\_CONTEXT spécifiées dans l'appel MQPUT ou MQPUT1 entraînent la génération par le gestionnaire de files d'attente des informations de contexte d'identité, le gestionnaire de files d'attente place *AlternateUserId* dans la zone *UserIdentifier* du MQMD du message, à la place de l'ID utilisateur en cours.

- Dans les autres environnements, *AlternateUserId* est utilisé uniquement pour les vérifications de contrôle d'accès sur l'objet ouvert. Si l'objet est une file d'attente, *AlternateUserId* n'affecte pas le contenu de la zone *UserIdentifier* dans le MQMD des messages envoyés à l'aide de cet identificateur de file d'attente.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par MQ\_USER\_ID\_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 12 caractères blancs dans les autres langages de programmation.

### **RecsPresent (MQLONG) pour MQOD**

Il s'agit du nombre d'enregistrements d'objet MQOR qui ont été fournis par l'application. Si ce nombre est supérieur à zéro, cela indique qu'une liste de distribution est en cours d'ouverture, *RecsPresent* étant le nombre de files d'attente de destination dans la liste. Une liste de distribution ne peut contenir qu'une seule destination.

La valeur de *RecsPresent* ne doit pas être inférieure à zéro, et si elle est supérieure à zéro, *ObjectType* doit être MQOT\_Q ; l'appel échoue avec le code anomalie MQRC\_RECS\_Présentat\_ERROR si ces conditions ne sont pas satisfaites.

Sous z/OS, cette zone doit être égale à zéro.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQOD\_VERSION\_2.

### **Nombre de KnownDest(MQLONG) pour MQOD**

Il s'agit du nombre de files d'attente de la liste de distribution qui ont été résolues en files d'attente locales et qui ont été ouvertes avec succès. Le nombre n'inclut pas les files d'attente qui se résolvent en files d'attente éloignées (même si une file d'attente de transmission locale est utilisée initialement pour stocker le message). Si cette zone est présente, elle est également définie lors de l'ouverture d'une file d'attente unique qui ne figure pas dans une liste de distribution.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQOD\_VERSION\_1.

### ***UnknownDest(MQLONG) pour MQOD***

Il s'agit du nombre de files d'attente de la liste de distribution qui sont résolues en files d'attente éloignées et qui ont été ouvertes avec succès. Si cette zone est présente, elle est également définie lors de l'ouverture d'une file d'attente unique qui ne figure pas dans une liste de distribution.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQOD\_VERSION\_1.

### ***InvalidDest(MQLONG) pour MQOD***

Il s'agit du nombre de files d'attente de la liste de distribution dont l'ouverture a échoué. Si cette zone est présente, elle est également définie lors de l'ouverture d'une file d'attente unique qui ne figure pas dans une liste de distribution.

**Remarque :** Si cette zone est présente, elle est définie uniquement si le paramètre **CompCode** de l'appel MQOPEN ou MQPUT1 est MQCC\_OK ou MQCC\_WARNING; elle n'est pas définie si le paramètre **CompCode** est MQCC\_FAILED.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQOD\_VERSION\_1.

### ***ObjectRecDécalage (MQLONG) pour MQOD***

Décalage en octets du premier enregistrement d'objet MQOR à partir du début de la structure MQOD. Le décalage peut être positif ou négatif. *ObjectRecOffset* est utilisé uniquement lorsqu'une liste de distribution est ouverte. La zone est ignorée si *RecsPresent* a la valeur zéro.

Lorsqu'une liste de distribution est ouverte, un tableau d'un ou de plusieurs enregistrements d'objet MQOR doit être fourni afin de spécifier les noms des files d'attente de destination dans la liste de distribution. Cette opération peut être effectuée de l'une des deux manières suivantes:

- En utilisant la zone de décalage *ObjectRecOffset*.

Dans ce cas, l'application doit déclarer sa propre structure contenant un MQOD suivi du tableau d'enregistrements MQOR (avec autant d'éléments de tableau que nécessaire) et définir *ObjectRecOffset* sur le décalage du premier élément du tableau à partir du début du MQOD. Assurez-vous que ce décalage est correct et qu'il comporte une valeur pouvant être prise en charge dans un MQLONG (le langage de programmation le plus restrictif est COBOL, pour lequel la plage valide est comprise entre -999 999 999 et +999 999 999).

Utilisez *ObjectRecOffset* pour les langages de programmation qui ne prennent pas en charge le type de données de pointeur ou qui implémentent le type de données de pointeur d'une manière qui n'est pas portable dans des environnements différents (par exemple, le langage de programmation COBOL).

- En utilisant la zone de pointeur *ObjectRecPtr*.

Dans ce cas, l'application peut déclarer le tableau de structures MQOR séparément de la structure MQOD et définir *ObjectRecPtr* sur l'adresse du tableau.

Utilisez *ObjectRecPtr* pour les langages de programmation qui prennent en charge le type de données de pointeur d'une manière portable dans différents environnements (par exemple, le langage de programmation C).

Quelle que soit la technique que vous choisissiez, utilisez l'une des options suivantes: *ObjectRecOffset* et *ObjectRecPtr* ; l'appel échoue avec le code anomalie MQRC\_OBJECT\_RECORDS\_ERROR si les deux valeurs sont nulles, ou les deux sont différentes de zéro.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQOD\_VERSION\_2.

### ***ResponseRecDécalage (MQLONG) pour MQOD***

Décalage en octets du premier enregistrement de réponse MQRR à partir du début de la structure MQOD. Le décalage peut être positif ou négatif. *ResponseRecOffset* est utilisé uniquement lorsqu'une liste de distribution est ouverte. La zone est ignorée si *RecsPresent* a la valeur zéro.

Lorsqu'une liste de distribution est ouverte, vous pouvez fournir un tableau d'un ou de plusieurs enregistrements de réponse MQRR afin d'identifier les files d'attente dont l'ouverture a échoué (*zoneCompCode* dans MQRR) et la raison de chaque échec (*zoneReason* dans MQRR). Les données sont renvoyées dans le tableau des enregistrements de réponse dans le même ordre que les noms de file d'attente dans le tableau des enregistrements d'objet. Le gestionnaire de files d'attente définit les enregistrements de réponse uniquement lorsque le résultat de l'appel est mixte (c'est-à-dire que certaines files d'attente ont été ouvertes avec succès alors que d'autres ont échoué, ou que toutes ont échoué, mais pour des raisons différentes) ; le code anomalie MQRC\_MULTIPLE\_MOTIFS de l'appel indique ce cas. Si le même code anomalie s'applique à toutes les files d'attente, ce code anomalie est renvoyé dans le paramètre **Reason** de l'appel MQOPEN ou MQPUT1 et les enregistrements de réponse ne sont pas définis. Les enregistrements de réponse sont facultatifs, mais s'ils sont fournis, ils doivent être *RecsPresent* .

Les enregistrements de réponse peuvent être fournis de la même manière que les enregistrements d'objet, soit en spécifiant un décalage dans *ResponseRecOffset*, soit en spécifiant une adresse dans *ResponseRecPtr* ; Pour plus d'informations sur la procédure à suivre, voir «[ObjectRecDécalage \(MQLONG\) pour MQOD](#)», à la page 511. Toutefois, *ResponseRecOffset* et *ResponseRecPtr* ne peuvent pas être utilisés ; l'appel échoue avec le code anomalie MQRC\_RESPONSE\_RECORDS\_ERROR si les deux sont différents de zéro.

Pour l'appel MQPUT1 , ces enregistrements de réponse sont utilisés pour renvoyer des informations sur les erreurs qui se produisent lorsque le message est envoyé aux files d'attente de la liste de distribution, ainsi que sur les erreurs qui se produisent lorsque les files d'attente sont ouvertes. Le code achèvement et le code anomalie de l'opération d'insertion pour une file d'attente remplacent ceux de l'opération d'ouverture pour cette file d'attente uniquement si le code achèvement de cette dernière était MQCC\_OK ou MQCC\_WARNING.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQOD\_VERSION\_2.

### ***ObjectRecPtr (MQPTR) pour MQOD***

Il s'agit de l'adresse du premier enregistrement d'objet MQOR. *ObjectRecPtr* est utilisé uniquement lorsqu'une liste de distribution est ouverte. La zone est ignorée si *RecsPresent* a la valeur zéro.

Vous pouvez utiliser *ObjectRecPtr* ou *ObjectRecOffset* pour spécifier les enregistrements d'objet, mais pas les deux ; pour la description de la zone *ObjectRecOffset* , voir «[ObjectRecDécalage \(MQLONG\) pour MQOD](#)», à la page 511. Si vous n'utilisez pas *ObjectRecPtr*, définissez-le sur le pointeur null ou sur les octets null.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire. Cette zone est ignorée si *Version* est inférieur à MQOD\_VERSION\_2.

**Remarque :** Sur les plateformes où le langage de programmation ne prend pas en charge le type de données pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée, la valeur initiale étant la chaîne d'octets tout-null.

### ***ResponseRecPtr (MQPTR) pour MQOD***



Il s'agit de l'adresse du premier enregistrement de réponse MQRR. *ResponseRecPtr* est utilisé uniquement lorsqu'une liste de distribution est ouverte. La zone est ignorée si *RecsPresent* a la valeur zéro.

Utilisez *ResponseRecPtr* ou *ResponseRecOffset* pour spécifier les enregistrements de réponse, mais pas les deux. Pour plus de détails, voir «*ResponseRecDécalage (MQLONG) pour MQOD*», à la page 512. Si vous n'utilisez pas *ResponseRecPtr*, définissez-le sur le pointeur null ou sur les octets null.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire. Cette zone est ignorée si *Version* est inférieur à MQOD\_VERSION\_2.

**Remarque :** Sur les plateformes où le langage de programmation ne prend pas en charge le type de données pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée, la valeur initiale étant la chaîne d'octets tout-null.

### **ID AlternateSecurity(MQBYTE40) pour MQOD**

Il s'agit d'un identificateur de sécurité qui est transmis avec *AlternateUserId* au service d'autorisation pour permettre l'exécution des vérifications d'autorisation appropriées. *AlternateSecurityId* est utilisé uniquement si :

- MQOO\_ALTERNATE\_USER\_AUTHORITY est spécifié dans l'appel MQOPEN, ou
- MQPMO\_ALTERNATE\_USER\_AUTHORITY est spécifié dans l'appel MQPUT1 ,

et la zone *AlternateUserId* n'est pas entièrement vide jusqu'au premier caractère null ou jusqu'à la fin de la zone.

Sous Windows, *AlternateSecurityId* peut être utilisé pour fournir l'identificateur de sécurité (SID) Windows qui identifie de manière unique le *AlternateUserId*. Le SID d'un utilisateur peut être obtenu à partir du système Windows à l'aide de l'appel API LookupAccountName () Windows .

Sous z/OS, cette zone est ignorée.

La zone *AlternateSecurityId* possède la structure suivante:

- Le premier octet est un entier binaire contenant la longueur des données significatives qui suivent ; la valeur exclut l'octet de longueur lui-même. Si aucun identificateur de sécurité n'est présent, la longueur est égale à zéro.
- Le deuxième octet indique le type d'identificateur de sécurité présent ; les valeurs suivantes sont possibles:

#### **ID\_SÉCURITÉ\_NT\_MQSIDT\_ID**

Identificateur de sécurité Windows .

#### **MQSIDT\_AUCUN**

Aucun identificateur de sécurité.

- Le troisième octet et les suivants jusqu'à la longueur définie par le premier octet contiennent l'identifiant de sécurité lui-même.
- Les octets restants dans la zone sont définis sur zéro binaire.

Vous pouvez utiliser la valeur spéciale suivante:

#### **MQSID\_AUCUN**

Aucun identificateur de sécurité n'a été indiqué.

La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQSID\_NONE\_ARRAY est également définie ; elle a la même valeur que MQSID\_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par MQ\_SECURITY\_ID\_LENGTH. La valeur initiale de cette zone est MQSID\_NONE. Cette zone est ignorée si *Version* est inférieur à MQOD\_VERSION\_3.

### ***ResolvedQName (MQCHAR48) pour MQOD***

Il s'agit du nom de la file d'attente de destination une fois que le gestionnaire de files d'attente local a résolu le nom. Le nom renvoyé est le nom d'une file d'attente qui existe sur le gestionnaire de files d'attente identifié par *ResolvedQMgrName*.

Une valeur non vide est renvoyée uniquement si l'objet est une file d'attente unique ouverte pour l'exploration, l'entrée ou la sortie (ou toute combinaison). Si l'objet ouvert est l'un des suivants, *ResolvedQName* est mis à blanc:

- N'est pas une file d'attente
- Une file d'attente, mais non ouverte pour l'exploration, l'entrée ou la sortie
- Une liste de distribution
- Une file d'attente alias qui fait référence à un objet de rubrique (voir [ResObjectString](#) à la place).
- File d'attente alias qui se résout en objet de rubrique.

Il s'agit d'une zone de sortie. La longueur de cette zone est indiquée par MQ\_Q\_NAME\_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation. Cette zone est ignorée si *Version* est inférieur à MQOD\_VERSION\_3.

### ***Nom ResolvedQMgr(MQCHAR48) pour MQOD***

Il s'agit du nom du gestionnaire de files d'attente de destination une fois que le gestionnaire de files d'attente local a résolu le nom. Le nom renvoyé est le nom du gestionnaire de files d'attente qui possède la file d'attente identifiée par *ResolvedQName*. *ResolvedQMgrName* peut être le nom du gestionnaire de files d'attente local.

Si *ResolvedQName* est une file d'attente partagée appartenant au groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local, *ResolvedQMgrName* est le nom du groupe de partage de files d'attente. Si la file d'attente appartient à un autre groupe de partage de files d'attente, *ResolvedQName* peut être le nom du groupe de partage de files d'attente ou le nom d'un gestionnaire de files d'attente membre du groupe de partage de files d'attente (la nature de la valeur renvoyée est déterminée par les définitions de files d'attente qui existent sur le gestionnaire de files d'attente local).

Une valeur non vide est renvoyée uniquement si l'objet est une file d'attente unique ouverte pour l'exploration, l'entrée ou la sortie (ou toute combinaison). Si l'objet ouvert est l'un des suivants, *ResolvedQMgrName* est mis à blanc:

- N'est pas une file d'attente
- Une file d'attente, mais non ouverte pour l'exploration, l'entrée ou la sortie
- Une file d'attente de cluster avec MQOO\_BIND\_NOT\_FIXED spécifiée (ou avec MQOO\_BIND\_AS\_Q\_DEF en vigueur lorsque l'attribut de file d'attente **DefBind** a la valeur MQBND\_BIND\_NOT\_FIXED)
- Une liste de distribution

Il s'agit d'une zone de sortie. La longueur de cette zone est indiquée par MQ\_Q\_NAME\_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation. Cette zone est ignorée si *Version* est inférieur à MQOD\_VERSION\_3.

### ***ObjectString (MQCHARV) pour MQOD***

La zone ObjectString indique le nom d'objet long.

Indique le nom d'objet long à utiliser. Cette zone n'est référencée que pour certaines valeurs de *ObjectType* et est ignorée pour toutes les autres valeurs. Voir la description de *ObjectType* pour plus de détails sur les valeurs qui indiquent que cette zone est utilisée.

Si *ObjectString* est spécifié de manière incorrecte, en fonction de la description de l'utilisation de la structure MQCHARV, ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie MQRC\_OBJECT\_STRING\_ERROR.

Il s'agit d'une zone d'entrée. Les valeurs initiales des zones de cette structure sont identiques à celles de la structure MQCHARV.

Le nom complet de la rubrique peut être généré à partir de deux zones différentes: *ObjectName* et *ObjectString*. Pour plus de détails sur l'utilisation de ces deux zones, voir [Combinaison de chaînes de rubrique](#).

### ***SelectionString (MQCHARV) pour MQOD***

Il s'agit de la chaîne utilisée pour fournir les critères de sélection utilisés lors de l'extraction de messages d'une file d'attente.

*SelectionString* ne doit pas être fourni dans les cas suivants:

- Si *ObjectType* n'est pas MQOT\_Q
- Si la file d'attente en cours d'ouverture n'est pas ouverte à l'aide de l'une des options MQOO\_BROWSE ou MQOO\_INPUT\_\*

Si *SelectionString* est fourni dans ces cas, l'appel échoue avec le code anomalie MQRC\_SELECTOR\_INVALID\_FOR\_TYPE.

Si *SelectionString* est spécifié de manière incorrecte, conformément à la description du mode d'utilisation de la structure «MQCHARV-Chaîne de longueur variable», à la page 303, ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie MQRC\_SELECTION\_STRING\_ERROR. La longueur maximale de *SelectionString* est MQ\_SELECTOR\_LENGTH.

L'utilisation de *SelectionString* est décrite dans [Sélecteurs](#).

### ***Chaîne ResObject(MQCHARV) pour MQOD***

La zone ResObjectString est le nom d'objet long après que le gestionnaire de files d'attente a résolu le nom fourni dans la zone *ObjectName*.

Cette zone est renvoyée uniquement pour les rubriques et les alias de file d'attente qui font référence à un objet de rubrique.

Si le nom d'objet long est fourni dans *ObjectString* et que rien n'est fourni dans *ObjectName*, la valeur renvoyée dans cette zone est la même que celle fournie dans *ObjectString*.

Si cette zone est omise (c'est-à-dire si ResObjectString.VSBufSize est égal à zéro), *ResObjectString* ne sera pas renvoyé, mais la longueur sera renvoyée dans ResObjectString.VSLength.

Si la longueur de la mémoire tampon (fournie dans ResObjectString.VSBufSize) est inférieure à la taille complète *ResObjectString*, la chaîne est tronquée et renvoie autant de caractères les plus à droite que possible dans la mémoire tampon fournie.

Si *ResObjectString* est spécifié de manière incorrecte, conformément à la description de l'utilisation de la structure MQCHARV, ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie MQRC\_RES\_OBJECT\_STRING\_ERROR.

### ***ResolvedType (MQLONG) pour MQOD***

Type de l'objet résolu (de base) en cours d'ouverture.

Les valeurs possibles sont les suivantes:

#### **MQOT\_Q**

L'objet résolu est une file d'attente. Cette valeur s'applique lorsqu'une file d'attente est ouverte directement ou lorsqu'une file d'attente alias pointant vers une file d'attente est ouverte.

#### **MQOT\_TOPIC**

L'objet résolu est une rubrique. Cette valeur s'applique lorsqu'une rubrique est ouverte directement ou lorsqu'une file d'attente alias pointant vers un objet de rubrique est ouverte.

#### **MQOT\_AUCUN**

Le type résolu n'est ni une file d'attente ni une rubrique.

## MQOR-Enregistrement d'objet

Utilisez la structure MQOR pour spécifier le nom de la file d'attente et le nom du gestionnaire de files d'attente d'une file d'attente de destination unique. MQOR est une structure d'entrée pour les appels MQOPEN et MQPUT1 .

### Disponibilité

La structure MQOR est disponible sur les plateformes suivantes:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour IBM MQ MQI clients connecté à ces systèmes.

### Jeu de caractères et codage

Les données dans MQOR doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et le codage du gestionnaire de files d'attente local indiqué par MQENC\_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

### Utilisation

En fournissant un tableau de ces structures sur l'appel MQOPEN, vous pouvez ouvrir une liste de files d'attente ; cette liste est appelée liste de distribution. Chaque insertion de message à l'aide de l'identificateur de file d'attente renvoyé par cet appel MQOPEN est placée dans chacune des files d'attente de la liste, à condition que l'ouverture de la file d'attente ait abouti.

### Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

<i>Tableau 505. Zones dans MQOR pour MQOR</i>		
<b>Nom et description de la zone</b>	<b>Nom de la constante</b>	<b>Valeur initiale (le cas échéant) de la constante</b>
<u>ObjectName</u> (nom de l'objet)	Aucun	Chaîne nulle ou blancs
<u>ObjectQMgrNom</u> (nom du gestionnaire de files d'attente d'objet)	Aucun	Chaîne nulle ou blancs
<b>Remarques :</b>		
1. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.		
2. Dans le langage de programmation C, la variable macroMQOR_DEFAULT contient les valeurs répertoriées dans la table. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:		
<pre>MQOR MyOR = {MQOR_DEFAULT};</pre>		

## Déclarations de langue

### Déclaration C pour MQOR

```
typedef struct tagMQOR MQOR;
struct tagMQOR {
    MQCHAR48  ObjectName;      /* Object name */
    MQCHAR48  ObjectQMgrName; /* Object queue manager name */
};
```

### Déclaration COBOL pour MQOR

```
** MQOR structure
10 MQOR.
** Object name
15 MQOR-OBJECTNAME PIC X(48).
** Object queue manager name
15 MQOR-OBJECTQMGRNAME PIC X(48).
```

### Déclaration PL/I pour MQOR

```
dcl
1 MQOR based,
3 ObjectName char(48), /* Object name */
3 ObjectQMgrName char(48); /* Object queue manager name */
```

### Déclaration Visual Basic pour MQOR

```
Type MQOR
    ObjectName As String*48 'Object name'
    ObjectQMgrName As String*48 'Object queue manager name'
End Type
```

### **ObjectName (MQCHAR48) pour MQOR**

Il s'agit de la même zone que la zone *ObjectName* dans la structure MQOD (voir MQOD pour plus de détails), sauf que:

- Il doit s'agir du nom d'une file d'attente.
- Il ne doit pas s'agir du nom d'une file d'attente modèle.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

### **ObjectQMgrNom (MQCHAR48) pour MQOR**

Il s'agit de la même zone que la zone *ObjectQMgrName* dans la structure MQOD (voir MQOD pour plus de détails).

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

## MQPD-Descripteur de propriété

La structure **MQPD** permet de définir les attributs d'une propriété. La structure est un paramètre d'entrée-sortie dans l'appel MQSETMP et un paramètre de sortie dans l'appel MQINQMP.

## Disponibilité

La structure **MQPD** est disponible sur les plateformes suivantes:

-  AIX

-  IBM i
-  Linux
-  Windows
-  z/OS

et pour IBM MQ MQI clients.

## Jeu de caractères et codage

Les données dans **MQPD** doivent être dans le jeu de caractères de l'application et dans le codage de l'application ( **MQENC\_NATIVE** ).

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

<i>Tableau 506. Zones dans MQPD</i>		
Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID_STRUC_MQPD_	'PD'
<u>Version</u> (numéro de version de la structure)	MQPD_VERSION_1	1
<u>Options</u> (options)	MQPD_AUCUN	0
<u>Support</u> (prise en charge requise pour la propriété de message)	MQPD_SUPPORT_XX_E NCODE_CASE_ONE facultatif	0
<u>Contexte</u> (contexte de message auquel appartient la propriété)	MQPD_NO_CONTEXT	0
<u>CopyOptions</u> (options de copie auxquelles la propriété appartient)	MQCOPY_DEFAULT	0
<p><b>Remarques :</b></p> <p>1. Dans le langage de programmation C, la variable de macro MQPD_DEFAULT contient les valeurs répertoriées dans la table. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</p> <pre>MQPD MyPD = {MQPD_DEFAULT};</pre>		

## Déclarations de langue

Déclaration C pour MQPD

```
typedef struct tagMQPD MQPD;
struct tagMQPD {
    MQCHAR4  StrucId;      /* Structure identifier */
    MQLONG   Version;     /* Structure version number */
    MQLONG   Options;     /* Options that control the action of
                          MQSETMP and MQINQMP */
    MQLONG   Support;     /* Property support option */
    MQLONG   Context;    /* Property context */
};
```

```

MQLONG CopyOptions; /* Property copy options */
};

```

## Déclaration COBOL pour MQPD

```

** MQPD structure
10 MQPD.
** Structure identifier
15 MQPD-STRUCID PIC X(4).
** Structure version number
15 MQPD-VERSION PIC S9(9) BINARY.
** Options that control the action of MQSETMP and
** MQINQMP
15 MQPD-OPTIONS PIC S9(9) BINARY.
** Property support option
15 MQPD-SUPPORT PIC S9(9) BINARY.
** Property context
15 MQPD-CONTEXT PIC S9(9) BINARY.
** Property copy options
15 MQPD-COPYOPTIONS PIC S9(9) BINARY.

```

## Déclaration PL/I pour MQPD

```

dcl
1 MQPD based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 Options fixed bin(31), /* Options that control the action
of MQSETMP and MQINQMP */
3 Support fixed bin(31), /* Property support option */
3 Context fixed bin(31), /* Property context */
3 CopyOptions fixed bin(31); /* Property copy options */

```

## Déclaration High Level Assembler pour MQPD

```

MQPD          DSECT
MQPD_STRUCID  DS   CL4   Structure identifier
MQPD_VERSION  DS   F     Structure version number
MQPD_OPTIONS  DS   F     Options that control the
*              action of MQSETMP and MQINQMP
MQPD_SUPPORT  DS   F     Property support option
MQPD_CONTEXT  DS   F     Property context
MQPD_COPYOPTIONS DS   F     Property copy options
MQPD_LENGTH   EQU  *-MQPD
MQPD_AREA     DS   CL(MQPD_LENGTH)

```

### **StrucId (MQCHAR4) pour MQPD**

Il s'agit de l'identificateur de structure de la structure de descripteur de propriété. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQPD\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUC\_MQD**

Identificateur de la structure de descripteur de propriété.

Pour le langage de programmation C, la constante **MQPD\_STRUC\_ID\_ARRAY** est également définie. Il a la même valeur que **MQPD\_STRUC\_ID**, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQPD**

Il s'agit du numéro de version de la structure ; la valeur doit être:

#### **MQPD\_VERSION\_1**

Structure de descripteur de propriété Version-1 .

La constante suivante indique le numéro de version de la version en cours:

## **MQPD\_CURRENT\_VERSION**

Version actuelle de la structure de descripteur de propriété.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **MQPD\_VERSION\_1**.

## **Options (MQLONG) pour MQPD**

La valeur doit être:

### **MQPD\_AUCUN**

Aucune option spécifiée

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQPD\_NONE.

## **Prise en charge (MQLONG) pour MQPD**

Cette zone indique le niveau de prise en charge de la propriété de message requis par le gestionnaire de files d'attente pour que le message contenant cette propriété soit inséré dans une file d'attente. S'applique uniquement aux propriétés définies par IBM MQ; la prise en charge de toutes les autres propriétés est facultative.

La zone est automatiquement définie sur la valeur correcte lorsque la propriété définie par IBM MQ est connue du gestionnaire de files d'attente. Si la propriété n'est pas reconnue, MQPD\_SUPPORT\_OPTIONAL est affectée. Lorsqu'un gestionnaire de files d'attente reçoit un message contenant une propriété définie par IBM MQ que le gestionnaire de files d'attente reconnaît comme étant incorrecte, il corrige la valeur de la zone *Support*.

Lors de la définition d'une propriété définie par IBM MQ à l'aide de l'appel MQSETMP sur un descripteur de message dans lequel l'option MQCMHO\_NO\_VALIDATION a été définie, *Support* devient une zone d'entrée. Cela permet à une application d'insérer une propriété définie par IBM MQ, avec la valeur correcte, dans laquelle la propriété n'est pas prise en charge par le gestionnaire de files d'attente connecté, mais dans laquelle le message doit être traité sur un autre gestionnaire de files d'attente.

La valeur MQPD\_SUPPORT\_OPTIONAL est toujours affectée aux propriétés qui ne sont pas des propriétés définies par IBM MQ.

Si un gestionnaire de files d'attente IBM WebSphere MQ 7.0, qui prend en charge les propriétés de message, reçoit une propriété qui contient une valeur *Support* non reconnue, la propriété est traitée comme si:

- MQPD\_SUPPORT\_REQUIRED a été spécifié si l'une des valeurs non reconnues est contenue dans MQPD\_REJECT\_UNSUP\_MASK.
- MQPD\_SUPPORT\_REQUIRED\_IF\_LOCAL a été spécifié si l'une des valeurs non reconnues est contenue dans le masque MQPD\_ACCEPT\_UNSUP\_IF\_XMIT\_MASK
- MQPD\_SUPPORT\_OPTIONAL a été spécifié dans le cas contraire.

L'une des valeurs suivantes est renvoyée par l'appel MQINQMP ou l'une des valeurs peut être spécifiée lors de l'utilisation de l'appel MQSETMP sur un descripteur de message dans lequel l'option MQCMHO\_NO\_VALIDATION est définie:

### **MQPD\_SUPPORT\_XX\_ENCODE\_CASE\_ONE facultatif**

La propriété est acceptée par un gestionnaire de files d'attente même si elle n'est pas prise en charge. La propriété peut être supprimée pour que le message soit transmis à un gestionnaire de files d'attente qui ne prend pas en charge les propriétés de message. Cette valeur est également affectée aux propriétés qui ne sont pas définies par IBM MQ.

### **MQPD\_SUPPORT\_REQUIS**

La prise en charge de la propriété est requise. Le message est rejeté par un gestionnaire de files d'attente qui ne prend pas en charge la propriété définie par IBM MQ. L'appel MQPUT ou MQPUT1 échoue avec le code achèvement MQCC\_FAILED et le code anomalie MQRC\_UNSUPPORTED\_PROPERTY.



## **MQPD\_SUPPORT\_REQUIRED\_IF\_LOCAL**

Le message est rejeté par un gestionnaire de files d'attente qui ne prend pas en charge la propriété définie par IBM MQ si le message est destiné à une file d'attente locale. L'appel MQPUT ou MQPUT1 échoue avec le code achèvement MQCC\_FAILED et le code anomalie MQRC\_UNSUPPORTED\_PROPERTY.

L'appel MQPUT ou MQPUT1 aboutit si le message est destiné à un gestionnaire de files d'attente éloignées.

Il s'agit d'une zone de sortie dans l'appel MQINQMP et d'une zone d'entrée dans l'appel MQSETMP si le descripteur de message a été créé avec l'option MQCMHO\_NO\_VALIDATION définie. La valeur initiale de cette zone est MQPD\_SUPPORT\_OPTIONAL.

## **Contexte (MQLONG) pour MQPD**

Décrit le contexte de message auquel appartient la propriété.

Lorsqu'un gestionnaire de files d'attente reçoit un message contenant une propriété définie par IBM MQ que le gestionnaire de files d'attente reconnaît comme étant incorrecte, il corrige la valeur de la zone *Context*.

L'option suivante peut être spécifiée:

### **CONTEXTE MQPD\_USER\_**

La propriété est associée au contexte utilisateur.

Aucune autorisation spéciale n'est requise pour pouvoir définir une propriété associée au contexte utilisateur à l'aide de l'appel MQSETMP.

Sur un gestionnaire de files d'attente IBM WebSphere MQ 7.0, une propriété associée au contexte utilisateur est sauvegardée comme décrit pour MQOO\_SAVE\_ALL\_CONTEXT. Un appel MQPUT avec MQPMO\_PASS\_ALL\_CONTEXT spécifié entraîne la copie de la propriété à partir du contexte sauvegardé dans le nouveau message.

Si l'option décrite précédemment n'est pas requise, l'option suivante peut être utilisée:

### **MQPD\_NO\_CONTEXT**

La propriété n'est pas associée à un contexte de message.

Une valeur non reconnue est rejetée avec le code *Reason* MQRC\_PD\_ERROR

Il s'agit d'une zone d'entrée-sortie de l'appel MQSETMP et d'une zone de sortie de l'appel MQINQMP. La valeur initiale de cette zone est MQPD\_NO\_CONTEXT.

## **CopyOptions (MQLONG) pour MQPD**

Décrit le type de message dans lequel la propriété doit être copiée. Il s'agit d'un champ de sortie uniquement pour les propriétés IBM MQ définies reconnues ; IBM MQ définit la valeur appropriée.

Lorsqu'un gestionnaire de files d'attente reçoit un message contenant une propriété définie par IBM MQ que le gestionnaire de files d'attente reconnaît comme étant incorrecte, il corrige la valeur de la zone *CopyOptions*.

Vous pouvez spécifier une ou plusieurs de ces options. Pour spécifier plusieurs options, ajoutez les valeurs ensemble (n'ajoutez pas la même constante plusieurs fois) ou combinez les valeurs à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations bit).

### **MQCOPY\_FORWARD**

Cette propriété est copiée dans un message en cours de transfert.

### **MQCOPY\_PUBLICATION**

Cette propriété est copiée dans le message reçu par un abonné lors de la publication d'un message.

### **REPONSE\_MQCOPY\_RÉPONSE**

Cette propriété est copiée dans un message de réponse.

### **RAPPORT MQCOPY\_REPORT**

Cette propriété est copiée dans un message de rapport.

## **MQCOPY\_ALL**

Cette propriété est copiée dans tous les types de messages suivants.

**Option par défaut:** L'option suivante peut être spécifiée pour fournir l'ensemble d'options de copie par défaut:

## **MQCOPY\_DEFAULT**

Cette propriété est copiée dans un message en cours de transfert, dans un message de rapport ou dans un message reçu par un abonné lors de la publication d'un message.

Cela revient à spécifier la combinaison des options MQCOPY\_FORWARD, MQCOPY\_REPORT et MQCOPY\_PUBLISH.

Si aucune des options décrites précédemment n'est requise, utilisez l'option suivante:

## **MQCOPY\_NONE**

Utilisez cette valeur pour indiquer qu'aucune autre option de copie n'est spécifiée ; à l'aide d'un programme, aucune relation n'existe entre cette propriété et les messages suivants. Cette valeur est toujours renvoyée pour les propriétés de descripteur de message.

Il s'agit d'une zone d'entrée-sortie de l'appel MQSETMP et d'une zone de sortie de l'appel MQINQMP. La valeur initiale de cette zone est MQCOPY\_DEFAULT.

## **MQPMO-Options de message d'insertion**

La structure MQPMO permet à l'application de spécifier des options qui contrôlent la façon dont les messages sont placés dans les files d'attente ou publiés dans les rubriques. La structure est un paramètre d'entrée-sortie dans les appels MQPUT et MQPUT1 .

## **Version**

La version actuelle de MQPMO est MQPMO\_VERSION\_3. Certaines zones sont disponibles uniquement dans certaines versions de MQPMO. Si vous devez porter des applications entre plusieurs environnements, vous devez vous assurer que la version de MQPMO est cohérente dans tous les environnements. Les zones qui n'existent que dans des versions particulières de la structure sont identifiées comme telles dans cette rubrique et dans les descriptions de zone.

Les fichiers d'en-tête, COPY et INCLUDE fournis pour les langages de programmation pris en charge contiennent la version la plus récente de MQPMO qui est prise en charge par l'environnement, mais avec la valeur initiale de la zone *Version* définie sur MQPMO\_VERSION\_1. Pour utiliser des zones qui ne sont pas présentes dans la structure version-1 , l'application doit définir la zone *Version* sur le numéro de version de la version requise.

## **Jeu de caractères et codage**

Les données dans MQPMO doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et le codage du gestionnaire de files d'attente local indiqué par MQENC\_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

## **Zones**

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

<i>Tableau 507. Zones dans MQPMO</i>		
<b>Nom et description de la zone</b>	<b>Nom de la constante</b>	<b>Valeur initiale (le cas échéant) de la constante</b>
StrucId (identificateur de structure)	ID_STRUC_MQPMO_	'PMO~'

Tableau 507. Zones dans MQPMO (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>Version</u> (numéro de version de la structure)	MQPMO_VERSION_1	1
<u>Options</u> (options qui contrôlent l'action de MQPUT et MQPUT1)	MQPMO_NONE	0
<u>Délai d'attente</u> (réservé)	Aucun	-1
<u>Contexte</u> (descripteur d'objet de la file d'entrée)	Aucun	0
<u>KnownDestNombre</u> (nombre de messages envoyés avec succès aux files d'attente locales)	Aucun	0
<u>UnknownDest</u> (nombre de messages envoyés avec succès aux files d'attente éloignées)	Aucun	0
<u>InvalidDestNombre</u> (nombre de messages qui n'ont pas pu être envoyés)	Aucun	0
<u>ResolvedQName</u> (nom résolu de la file d'attente de destination)	Aucun	Chaîne nulle ou blancs
<u>ResolvedQMgrNom</u> (nom résolu du gestionnaire de files d'attente de destination)	Aucun	Chaîne nulle ou blancs
<b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieur à MQPMO_VERSION_2.		
<u>RecsPresent</u> (nombre d'enregistrements de message d'insertion ou d'enregistrements de réponse présents)	Aucun	0
<u>PutMsgRecFields</u> (indicateurs indiquant les zones MQPMR qui sont présentes)	MQPMRF_AUCUN	0
<u>PutMsgRecOffset</u> (décalage du premier enregistrement de message d'insertion depuis le démarrage de MQPMO)	Aucun	0
<u>ResponseRecOffset</u> (décalage du premier enregistrement de réponse à partir du démarrage de MQPMO)	Aucun	0
<u>PutMsgRecPtr</u> (adresse du premier enregistrement de message d'insertion)	Aucun	Pointeur null ou octets null
<u>ResponseRecPtr</u> (adresse du premier enregistrement de réponse)	Aucun	Pointeur null ou octets null
<b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieur à MQPMO_VERSION_3.		
<u>OriginalMsgDescripteur</u> (descripteur de message d'origine)	MQHM_NONE	0
<u>NewMsgDescripteur</u> (nouveau descripteur de message)	MQHM_NONE	0
<u>Action</u> (type d'insertion effectuée et relation entre le message d'origine spécifié par la zone <i>OriginalMsgHandle</i> et le nouveau message spécifié par la zone <i>NewMsgHandle</i> )	MQACTP_NOUVEAU	0

Tableau 507. Zones dans MQPMO (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
PubLevel (niveau d'abonnement ciblé par la publication)	Aucun	9
<p><b>Remarques :</b></p> <ol style="list-style-type: none"> <li>1. Le symbole ↵ représente un caractère blanc unique.</li> <li>2. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.</li> <li>3. Dans le langage de programmation C, la variable macroMQPMO_DEFAULT contient les valeurs répertoriées dans la table. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</li> </ol> <pre>MQPMO MyPMO = {MQPMO_DEFAULT};</pre>		

## Déclarations de langue

### Déclaration C pour MQPMO

```
typedef struct tagMQPMO MQPMO;
struct tagMQPMO {
    MQCHAR4    StructId;           /* Structure identifier */
    MQLONG     Version;           /* Structure version number */
    MQLONG     Options;           /* Options that control the action of
    MQPUT and MQPUT1 */

    MQLONG     Timeout;           /* Reserved */
    MQHOBJS    Context;           /* Object handle of input queue */
    MQLONG     KnownDestCount;    /* Number of messages sent
    successfully to local queues */
    MQLONG     UnknownDestCount;  /* Number of messages sent
    successfully to remote queues */
    MQLONG     InvalidDestCount;  /* Number of messages that could not
    be sent */
    MQCHAR48   ResolvedQName;     /* Resolved name of destination
    queue */
    MQCHAR48   ResolvedQMgrName;  /* Resolved name of destination queue
    manager */

    /* Ver:1 */
    MQLONG     RecsPresent;        /* Number of put message records or
    response records present */
    MQLONG     PutMsgRecFields;    /* Flags indicating which MQPMR fields
    are present */
    MQLONG     PutMsgRecOffset;    /* Offset of first put message record
    from start of MQPMO */
    MQLONG     ResponseRecOffset; /* Offset of first response record
    from start of MQPMO */
    MQPTR      PutMsgRecPtr;       /* Address of first put message
    record */
    MQPTR      ResponseRecPtr;     /* Address of first response record */

    /* Ver:2 */
    MQHMSG     OriginalMsgHandle;  /* Original message handle */
    MQHMSG     NewMsgHandle;       /* New message handle */
    MQLONG     Action;             /* The action being performed */
    MQLONG     PubLevel;           /* Subscription level */

    /* Ver:3 */
};
```

### Déclaration COBOL pour MQPMO

```
** MQPMO structure
   10 MQPMO.
** Structure identifier
```

```

15 MQPMO-STRUCID          PIC X(4).
** Structure version number
15 MQPMO-VERSION          PIC S9(9) BINARY.
** Options that control the action of MQPUT and MQPUT1
15 MQPMO-OPTIONS          PIC S9(9) BINARY.
** Reserved
15 MQPMO-TIMEOUT          PIC S9(9) BINARY.
** Object handle of input queue
15 MQPMO-CONTEXT          PIC S9(9) BINARY.
** Number of messages sent successfully to local queues
15 MQPMO-KNOWNDSTCOUNT  PIC S9(9) BINARY.
** Number of messages sent successfully to remote queues
15 MQPMO-UNKNOWNDSTCOUNT PIC S9(9) BINARY.
** Number of messages that could not be sent
15 MQPMO-INVALIDDSTCOUNT PIC S9(9) BINARY.
** Resolved name of destination queue
15 MQPMO-RESOLVEDQNAME    PIC X(48).
** Resolved name of destination queue manager
15 MQPMO-RESOLVEDQMGRNAME PIC X(48).
** Number of put message records or response records present
15 MQPMO-RECSPRESENT     PIC S9(9) BINARY.
** Flags indicating which MQPMR fields are present
15 MQPMO-PUTMSGRECFIELDS PIC S9(9) BINARY.
** Offset of first put message record from start of MQPMO
15 MQPMO-PUTMSGRECOFFSET PIC S9(9) BINARY.
** Offset of first response record from start of MQPMO
15 MQPMO-RESPONSERECOFFSET PIC S9(9) BINARY.
** Address of first put message record
15 MQPMO-PUTMSGRECPTTR   POINTER.
** Address of first response record
15 MQPMO-RESPONSERECPTTR POINTER.
** Original message handle
15 MQPMO-ORIGINALMSGHANDLE PIC S9(18) BINARY.
** New message handle
15 MQPMO-NEWMMSGHANDLE    PIC S9(18) BINARY.
** The action being performed
15 MQPMO-ACTION           PIC S9(9) BINARY.
** Publish level
15 MQPMO-PUBLEVEL        PIC S9(9) BINARY.

```

## Déclaration PL/I pour MQPMO

```

dcl
1 MQPMO based,
3 StructId          char(4),          /* Structure identifier */
3 Version           fixed bin(31),    /* Structure version number */
3 Options           fixed bin(31),    /* Options that control the action
of MQPUT and MQPUT1 */
3 Timeout          fixed bin(31),    /* Reserved */
3 Context          fixed bin(31),    /* Object handle of input queue */
3 KnownDestCount   fixed bin(31),    /* Number of messages sent
successfully to local queues */
3 UnknownDestCount fixed bin(31),    /* Number of messages sent
successfully to remote queues */
3 InvalidDestCount fixed bin(31),    /* Number of messages that could
not be sent */
3 ResolvedQName     char(48),        /* Resolved name of destination
queue */
3 ResolvedQMgrName char(48),        /* Resolved name of destination
queue manager */
3 RecsPresent       fixed bin(31),    /* Number of put message records or
response records present */
3 PutMsgRecFields   fixed bin(31),    /* Flags indicating which MQPMR
fields are present */
3 PutMsgRecOffset   fixed bin(31),    /* Offset of first put message
record from start of MQPMO */
3 ResponseRecOffset fixed bin(31),    /* Offset of first response record
from start of MQPMO */
3 PutMsgRecPtr      pointer,         /* Address of first put message
record */
3 ResponseRecPtr    pointer,         /* Address of first response
record */
3 OriginalMsgHandle fixed bin(63),    /* Original message handle */
3 NewMsgHandle      fixed bin(63);    /* New message handle */
3 Action            fixed bin(31);    /* The action being performed */
3 PubLevel          fixed bin(31);    /* Publish level */

```

## Déclaration High Level Assembler pour MQPMO

```

MQPMO                DSECT
MQPMO_STRUCID        DS   CL4   Structure identifier
MQPMO_VERSION        DS   F     Structure version number
MQPMO_OPTIONS        DS   F     Options that control the action of
*                    *         MQPUT and MQPUT1
MQPMO_TIMEOUT        DS   F     Reserved
MQPMO_CONTEXT        DS   F     Object handle of input queue
MQPMO_KNOWNDESTCOUNT DS   F     Number of messages sent successfully
*                    *         to local queues
MQPMO_UNKNOWNDSTCOUNT DS   F     Number of messages sent successfully
*                    *         to remote queues
MQPMO_INVALIDDESTCOUNT DS   F     Number of messages that could not be
*                    *         sent
MQPMO_RESOLVEDQNAME  DS   CL48  Resolved name of destination queue
MQPMO_RESOLVEDQMGRNAME DS   CL48 Resolved name of destination queue
*                    *         manager
MQPMO_RECSPRESENT    DS   F     Number of put message records or
*                    *         response records present
MQPMO_PUTMSGRECFIELDS DS   F     Flags indicating which MQPMR
*                    *         fields are present
MQPMO_PUTMSGRECOFFSET DS   F     Offset of first put message record
*                    *         from start of MQPMO
MQPMO_RESPONSERECOFFSET DS   F     Offset of first response record
*                    *         from start of MQPMO
MQPMO_PUTMSGRECPtr    DS   F     Address of first put message
*                    *         record
MQPMO_RESPONSERECPtr  DS   F     Address of first response record
MQPMO_ORIGINALMSGHANDLE DS   D     Original message handle
MQPMO_NEWMSGHANDLE    DS   D     New message handle
MQPMO_ACTION         DS   F     The action being performed
MQPMO_PUBLEVEL       DS   F     Publish level
*
MQPMO_LENGTH         EQU   *-MQPMO
                     ORG   MQPMO
MQPMO_AREA           DS    CL(MQPMO_LENGTH)

```

## Déclaration Visual Basic pour MQPMO

```

Type MQPMO
  StrucId      As String*4   'Structure identifier'
  Version      As Long       'Structure version number'
  Options      As Long       'Options that control the action of
                             'MQPUT and MQPUT1'
  Timeout      As Long       'Reserved'
  Context      As Long       'Object handle of input queue'
  KnownDestCount As Long     'Number of messages sent successfully
                             'to local queues'
  UnknownDestCount As Long   'Number of messages sent successfully
                             'to remote queues'
  InvalidDestCount As Long   'Number of messages that could not be
                             'sent'
  ResolvedQName As String*48 'Resolved name of destination queue'
  ResolvedQMgrName As String*48 'Resolved name of destination queue
                             'manager'
  RecsPresent   As Long       'Number of put message records or
                             'response records present'
  PutMsgRecFields As Long     'Flags indicating which MQPMR fields
                             'are present'
  PutMsgRecOffset As Long     'Offset of first put message record
                             'from start of MQPMO'
  ResponseRecOffset As Long   'Offset of first response record from
                             'start of MQPMO'
  PutMsgRecPtr  As MQPTR     'Address of first put message record'
  ResponseRecPtr As MQPTR     'Address of first response record'
End Type

```

### **StrucId (MQCHAR4) pour MQPMO**

Il s'agit de l'identificateur de structure de la structure des options de message d'insertion. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQPMO\_STRUC\_ID.

La valeur doit être:

## **ID\_STRUC\_MQPMO\_**

Identificateur de la structure des options de message d'insertion.

Pour le langage de programmation C, la constante MQPMO\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQPMO\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

## **Version (MQLONG) pour MQPMO**

Numéro de version de la structure.

Il doit s'agir de l'une des valeurs suivantes :

### **MQPMO\_VERSION\_1**

Structure des options d'insertion de message Version-1 .

Cette version est prise en charge dans tous les environnements.

### **MQPMO\_VERSION\_2**

Structure des options d'insertion de message Version-2 .

Cette version est prise en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour IBM MQ MQI clients connecté à ces systèmes.

### **MQPMO\_VERSION\_3**

Structure des options d'insertion de message Version-3 .

Cette version est prise en charge dans tous les environnements.

Les zones qui existent uniquement dans la version la plus récente de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

### **MQPMO\_CURRENT\_VERSION**

Version actuelle de la structure des options d'insertion de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQPMO\_VERSION\_1.

## **Options (MQLONG) pour MQPMO**

La zone Options contrôle le fonctionnement des appels **MQPUT** et **MQPUT1** .

**Option de portée.** Vous pouvez spécifier l'une ou l'autre des options MQPMO. Pour spécifier plusieurs options, ajoutez les valeurs ensemble (n'ajoutez pas la même constante plusieurs fois) ou combinez les valeurs à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations bit). Les combinaisons qui ne sont pas valides sont notées ; toutes les autres combinaisons sont valides.

L'option suivante contrôle la portée des publications envoyées:

### **MQPMO\_SCOPE\_QMGR**

La publication est envoyée uniquement aux abonnés qui se sont abonnés à ce gestionnaire de files d'attente. La publication n'est pas réacheminée vers les gestionnaires de files d'attente de publication / abonnement distants qui ont souscrit un abonnement à ce gestionnaire de files d'attente, ce qui remplace tout comportement défini à l'aide de l'attribut de rubrique PUBSCOPE.

**Remarque :** S'il n'est pas défini, la portée de la publication est déterminée par l'attribut de rubrique PUBSCOPE.

**Options de publication.** Les options suivantes contrôlent le mode de publication des messages dans une rubrique:

### **MQPMO\_SUPPRESS\_REPLYTO**

Les informations indiquées dans les zones *ReplyToQ* et *ReplyToQMgr* du MQMD de cette publication ne sont pas transmises aux abonnés. Si cette option est utilisée avec une option de rapport qui requiert un *ReplyToQ*, l'appel échoue avec MQRC\_MISSING\_REPLY\_TO\_Q.

### **MQPMO\_RETAIN**

La publication envoyée doit être conservée par le gestionnaire de files d'attente. Cette conservation permet à un abonné de demander une copie de cette publication après sa publication, à l'aide de l'appel MQSUBRQ. Elle permet également d'envoyer une publication aux applications qui effectuent leur abonnement après la date de cette publication (sauf si elles choisissent de ne pas l'envoyer à l'aide de l'option MQSO\_NEW\_PUBLICATIONS\_ONLY). Si une application reçoit une publication qui a été conservée, elle est indiquée par la propriété de message MQIsRetained de cette publication.

Une seule publication peut être conservée sur chaque noeud de l'arborescence de rubriques. Par conséquent, s'il existe déjà une publication conservée pour cette rubrique, publiée par toute autre application, elle est remplacée par cette publication. Il est donc préférable d'éviter que plus d'un diffuseur de publications conserve des messages sur le même sujet.

Lorsque des publications conservées sont demandées par un abonné, l'abonnement utilisé peut contenir un caractère générique dans la rubrique, auquel cas un certain nombre de publications conservées peuvent correspondre (sur différents noeuds de l'arborescence de rubriques) et plusieurs publications peuvent être envoyées à l'application demandeuse. Pour plus de détails, voir la description de l'appel «MQSUBRQ-Demande d'abonnement», à la page 832 .

Pour plus d'informations sur la façon dont les publications conservées interagissent avec les niveaux d'abonnement, voir [Interception des publications](#).

Si cette option est utilisée et que la publication ne peut pas être conservée, le message n'est pas publié et l'appel échoue avec MQRC\_PUT\_NOT\_RETAINED.

### **MQPMO\_NOT\_OWN\_SUBS**

Indique au gestionnaire de files d'attente que l'application ne souhaite envoyer aucune de ses publications aux abonnements qu'elle possède. Les abonnements sont considérés comme appartenant à la même application si les descripteurs de connexion sont identiques.

### **MQPMO\_WARN\_SI\_NO\_SUBS\_MIS en correspondance**

Si aucun abonnement ne correspond à la publication, renvoyez un code achèvement (*CompCode*) de MQCC\_WARNING et le code anomalie MQRC\_NO\_SUBS\_MATCH.

Si MQRC\_NO\_SUBS\_APPARIÉ est renvoyé par l'opération d'insertion, la publication n'a été distribuée à aucun abonnement. Toutefois, si l'option MQPMO\_RETAIN est spécifiée sur l'opération d'insertion, le message est conservé et distribué à tout abonnement correspondant défini ultérieurement.

Un abonnement à la rubrique correspond à la publication si l'une des conditions suivantes est remplie:

- Le message est distribué dans la file d'attente d'abonnement
- Le message aurait été distribué à la file d'attente d'abonnement, mais un problème avec la file d'attente signifie que le message ne peut pas être placé dans la file d'attente et qu'il a donc été placé dans la file d'attente de rebut ou supprimé.
- Un exit de routage est défini pour supprimer la distribution du message à l'abonnement

Un abonnement à la rubrique ne correspond pas à la publication si l'une des conditions suivantes est remplie:

- L'abonnement comporte une chaîne de sélection qui ne correspond pas à la publication
- L'abonnement a spécifié l'option MQSO\_PUBLICATION\_ON\_REQUEST
- La publication n'est pas distribuée car l'option MQPMO\_NOT\_OWN\_SUBS a été spécifiée lors de l'opération d'insertion et l'abonnement correspond à l'identité du diffuseur de publications



**Options de point de synchronisation.** Les options suivantes concernent la participation de l'appel MQPUT ou MQPUT1 dans une unité d'oeuvre:

### **MQPMO\_SYNCPOINT**

La demande consiste à fonctionner dans le cadre des protocoles d'unité de travail normaux. Le message n'est pas visible en dehors de l'unité de travail tant que l'unité de travail n'est pas validée. Si l'unité d'oeuvre est annulée, le message est supprimé.

Si MQPMO\_SYNCPOINT et MQPMO\_NO\_SYNCPOINT ne sont pas spécifiés, l'inclusion de la demande d'insertion dans les protocoles d'unité de travail est déterminée par l'environnement exécutant le gestionnaire de files d'attente et non par l'environnement exécutant l'application. Sous z/OS, la demande d'insertion se trouve dans une unité de travail. Dans tous les autres environnements, la demande d'insertion ne se trouve pas dans une unité de travail.

En raison de ces différences, une application que vous souhaitez porter ne doit pas autoriser cette option à utiliser par défaut ; indiquez MQPMO\_SYNCPOINT ou MQPMO\_NO\_SYNCPOINT de manière explicite.

N'indiquez pas MQPMO\_SYNCPOINT avec MQPMO\_NO\_SYNCPOINT.

### **MQPMO\_NO\_SYNCPOINT**

La demande consiste à fonctionner en dehors des protocoles d'unité de travail normaux. Le message est disponible immédiatement et ne peut pas être supprimé par l'annulation d'une unité de travail.

Si MQPMO\_NO\_SYNCPOINT et MQPMO\_SYNCPOINT ne sont pas spécifiés, l'inclusion de la demande d'insertion dans les protocoles d'unité de travail est déterminée par l'environnement exécutant le gestionnaire de files d'attente et non par l'environnement exécutant l'application. Sous z/OS, la demande d'insertion se trouve dans une unité de travail. Dans tous les autres environnements, la demande d'insertion ne se trouve pas dans une unité de travail.

En raison de ces différences, une application que vous souhaitez porter ne doit pas autoriser cette option à utiliser par défaut ; indiquez MQPMO\_SYNCPOINT ou MQPMO\_NO\_SYNCPOINT de manière explicite.

Ne spécifiez pas MQPMO\_NO\_SYNCPOINT avec MQPMO\_SYNCPOINT.

**Options d'identificateur de message et d'identificateur de corrélation.** Les options suivantes demandent au gestionnaire de files d'attente de générer un nouvel identificateur de message ou un nouvel identificateur de corrélation:

### **ID MQPMO\_NEW\_MSG\_ID**

Le gestionnaire de files d'attente remplace le contenu de la zone *MsgId* dans MQMD par un nouvel identificateur de message. Cet identificateur de message est envoyé avec le message et renvoyé à l'application dans la sortie de l'appel MQPUT ou MQPUT1 .

L'option MQPMO\_NEW\_MSG\_ID peut également être spécifiée lorsque le message est inséré dans une liste de distribution. Pour plus de détails, voir la description de la zone *MsgId* dans la structure MQPMR.

L'utilisation de cette option évite à l'application de devoir réinitialiser la zone *MsgId* sur MQMI\_NONE avant chaque appel MQPUT ou MQPUT1 .

### **MQPMO\_NEW\_CORREL\_ID**

Le gestionnaire de files d'attente remplace le contenu de la zone *CorrelId* dans MQMD par un nouvel identificateur de corrélation. Cet identificateur de corrélation est envoyé avec le message et renvoyé à l'application en sortie de l'appel MQPUT ou MQPUT1 .

L'option MQPMO\_NEW\_CORREL\_ID peut également être spécifiée lorsque le message est inséré dans une liste de distribution. Pour plus de détails, voir la description de la zone *CorrelId* dans la structure MQPMR.

MQPMO\_NEW\_CORREL\_ID est utile lorsque l'application requiert un identificateur de corrélation unique.

**Options de groupe et de segment.** Les options suivantes concernent le traitement des messages dans des groupes et des segments de messages logiques. Lisez les définitions qui suivent pour vous aider à comprendre l'option.



**Avertissement :** Vous ne pouvez pas utiliser de messages segmentés ou groupés avec la fonction de publication / abonnement.

### Message physique

Il s'agit de la plus petite unité d'informations pouvant être placée dans une file d'attente ou supprimée d'une file d'attente ; elle correspond souvent aux informations spécifiées ou extraites sur un seul appel MQPUT, MQPUT1 ou MQGET. Chaque message physique possède son propre descripteur de message (MQMD). En règle générale, les messages physiques se distinguent par des valeurs différentes pour l'identificateur de message (*zoneMsgId* dans MQMD), bien que cela ne soit pas appliqué par le gestionnaire de files d'attente.

### Message logique

Un message logique est une unité unique d'informations d'application pour les plateformes non z/OS uniquement. En l'absence de contraintes système, un message logique est identique à un message physique. Toutefois, lorsque les messages logiques sont extrêmement volumineux, les contraintes du système peuvent rendre souhaitable ou nécessaire la division d'un message logique en deux ou plusieurs messages physiques, appelés *segments*.

Un message logique segmenté se compose d'au moins deux messages physiques ayant le même identificateur de groupe non nul (*zoneGroupId* dans MQMD) et le même numéro de séquence de message (*zoneMsgSeqNumber* dans MQMD). Les segments se distinguent par des valeurs différentes pour le décalage de segment (*zoneOffset* dans MQMD), qui fournit le décalage des données dans le message physique à partir du début des données dans le message logique. Chaque segment étant un message physique, les segments d'un message logique ont généralement des identificateurs de message différents.

Un message logique qui n'a pas été segmenté, mais pour lequel la segmentation a été autorisée par l'application émettrice, possède également un identificateur de groupe non nul, bien que dans ce cas, il n'y ait qu'un seul message physique avec cet identificateur de groupe si le message logique n'appartient pas à un groupe de messages. Les messages logiques pour lesquels la segmentation a été inhibée par l'application émettrice ont un identificateur de groupe nul (MQGI\_NONE), sauf si le message logique appartient à un groupe de messages.

### Groupe de messages

Un groupe de messages est un ensemble d'un ou de plusieurs messages logiques ayant le même identificateur de groupe non nul. Les messages logiques du groupe se distinguent par des valeurs différentes pour le numéro de séquence de message, qui est un entier compris entre 1 et  $n$ , où  $n$  est le nombre de messages logiques du groupe. Si un ou plusieurs messages logiques sont segmentés, il y a plus de  $n$  messages physiques dans le groupe.

### MQPMO\_LOGICAL\_ORDER

Cette option indique au gestionnaire de files d'attente comment l'application insère des messages dans des groupes et des segments de messages logiques. Elle ne peut être spécifiée que dans l'appel MQPUT ; elle n'est pas valide dans l'appel MQPUT1 .

Si MQPMO\_LOGICAL\_ORDER est spécifié, il indique que l'application utilise des appels MQPUT successifs pour :

1. Placer les segments dans chaque message logique dans l'ordre croissant des décalages de segment, à partir de 0, sans écart.
2. Placer tous les segments dans un message logique avant de les placer dans le message logique suivant.
3. Placer les messages logiques dans chaque groupe de messages dans l'ordre croissant des numéros de séquence de message, à partir de 1, sans écart. IBM MQ incrémente le numéro de séquence de message automatiquement.
4. Placer tous les messages logiques dans un groupe de messages avant de les placer dans le groupe de messages suivant.

Pour plus d'informations sur MQPMO\_LOGICAL\_ORDER, voir [Ordre logique et physique](#)

**Options de contexte.** Les options suivantes contrôlent le traitement du contexte de message:

#### **MQPMO\_NO\_CONTEXT**

Le contexte d'identité et le contexte d'origine sont définis pour indiquer l'absence de contexte. Cela signifie que les zones de contexte dans MQMD sont définies sur:

- Blancs pour les zones de caractères
- Valeurs nulles pour les zones d'octet
- Zéros pour les champs numériques

#### **MQPMO\_CONTEXTE\_PAR\_DÉFAUT**

Le message doit être associé à des informations de contexte par défaut, à la fois pour l'identité et l'origine. Le gestionnaire de files d'attente définit les zones de contexte dans le descripteur de message comme suit:

Tableau 508. Valeurs d'informations contextuelles par défaut pour les zones MQMD

<b>Zone dans MQMD</b>	<b>Valeur utilisée</b>
<i>UserIdentifier</i>	Déterminé à partir de l'environnement si possible ; sinon, il est mis à blanc.
<i>AccountingToken</i>	Déterminé à partir de l'environnement si possible ; défini sur MQACT_NONE dans le cas contraire.
<i>AppIdentityData</i>	Mettez à blanc.
<i>PutAppType</i>	Déterminé à partir de l'environnement.
<i>PutAppName</i>	Déterminé à partir de l'environnement si possible ; sinon, il est mis à blanc.
<i>PutDate</i>	Définissez la date à laquelle le message est inséré.
<i>PutTime</i>	Défini sur l'heure à laquelle le message est inséré.
<i>AppOriginData</i>	Mettez à blanc.

Pour plus d'informations sur le contexte de message, voir [Contexte de message](#).

Il s'agit des valeurs par défaut et des actions si aucune option de contexte n'est spécifiée.

#### **MQPMO\_PASS\_IDENTITY\_CONTEXT**

Le message doit être associé à des informations de contexte. Le contexte d'identité est extrait de l'identificateur de file d'attente spécifié dans la zone *Context*. Les informations de contexte d'origine sont générées par le gestionnaire de files d'attente de la même manière que pour MQPMO\_DEFAULT\_CONTEXT (voir le tableau précédent pour les valeurs). Pour plus d'informations sur le contexte de message, voir [Contexte de message](#).

Pour l'appel MQPUT, la file d'attente doit avoir été ouverte avec l'option MQOO\_PASS\_IDENTITY\_CONTEXT (ou une option qui l'implique). Pour l'appel MQPUT1, la même vérification d'autorisation est effectuée que pour l'appel MQOPEN avec l'option MQOO\_PASS\_IDENTITY\_CONTEXT.

#### **MQPMO\_PASS\_ALL\_CONTEXT**

Le message doit être associé à des informations de contexte. Le contexte est extrait de l'identificateur de file d'attente spécifié dans la zone *Context*. Pour plus d'informations sur le contexte de message, voir [Contrôle des informations de contexte](#).

Pour l'appel MQPUT, la file d'attente doit avoir été ouverte avec l'option MQOO\_PASS\_ALL\_CONTEXT (ou une option qui l'implique). Pour l'appel MQPUT1, la même vérification d'autorisation est effectuée que pour l'appel MQOPEN avec l'option MQOO\_PASS\_ALL\_CONTEXT.

### **MQPMO\_SET\_IDENTITY\_CONTEXT**

Le message doit être associé à des informations de contexte. L'application spécifie le contexte d'identité dans la structure MQMD. Les informations de contexte d'origine sont générées par le gestionnaire de files d'attente de la même manière que pour MQPMO\_DEFAULT\_CONTEXT (voir le tableau précédent pour les valeurs). Pour plus d'informations sur le contexte de message, voir [Contexte de message](#).

Pour l'appel MQPUT, la file d'attente doit avoir été ouverte avec l'option MQOO\_SET\_IDENTITY\_CONTEXT (ou une option qui l'implique). Pour l'appel MQPUT1, la même vérification d'autorisation est effectuée que pour l'appel MQOPEN avec l'option MQOO\_SET\_IDENTITY\_CONTEXT.

### **MQPMO\_SET\_ALL\_CONTEXT**

Le message doit être associé à des informations de contexte. L'application spécifie l'identité, l'origine et le contexte utilisateur dans la structure MQMD. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#).

Pour l'appel MQPUT, la file d'attente doit avoir été ouverte avec l'option MQOO\_SET\_ALL\_CONTEXT. Pour l'appel MQPUT1, la même vérification d'autorisation est effectuée que pour l'appel MQOPEN avec l'option MQOO\_SET\_ALL\_CONTEXT.

Vous ne pouvez spécifier qu'une seule des options de contexte MQPMO\_\*\_CONTEXT. Si vous n'en spécifiez aucun, MQPMO\_DEFAULT\_CONTEXT est pris en compte.

**Options de propriété.** L'option suivante concerne les propriétés du message:

### **MQPMO\_MD\_FOR\_OUTPUT\_ONLY**

Le paramètre de descripteur de message ne doit être utilisé que pour la sortie afin de renvoyer le descripteur de message du message qui a été inséré. Les zones de descripteur de message associées aux zones *NewMsgHandle*, *OriginalMsgHandle*, ou aux deux, de la structure **MQPMO** doivent être utilisées pour l'entrée.

Si aucun descripteur de message valide n'est fourni, l'appel échoue avec le code anomalie **MQRC\_MD\_ERROR**.

**Options de réponse d'insertion.** Les options suivantes contrôlent la réponse renvoyée à un appel MQPUT ou MQPUT1. Vous ne pouvez spécifier qu'une seule de ces options. Si MQPMO\_ASYNC\_RESPONSE et MQPMO\_SYNC\_RESPONSE ne sont pas spécifiés, MQPMO\_RESPONSE\_AS\_Q\_DEF ou MQPMO\_RESPONSE\_AS\_TOPIC\_DEF est utilisé.

### **MQPMO\_ASYNC\_RESPONSE**

L'option MQPMO\_ASYNC\_RESPONSE demande qu'une opération MQPUT ou MQPUT1 soit terminée sans que l'application attende que le gestionnaire de files d'attente termine l'appel. L'utilisation de cette option permet d'améliorer les performances de la messagerie, en particulier pour les applications utilisant des liaisons client. Une application peut vérifier périodiquement, à l'aide de l'instruction MQSTAT, si une erreur s'est produite lors des appels asynchrones précédents.

Avec cette option, seules les zones suivantes sont garanties pour être renseignées dans le MQMD ;

- ApplIdentityData
- PutApplType
- PutApplName
- ApplOriginData

En outre, si l'un des MQPMO\_NEW\_MSG\_ID ou MQPMO\_NEW\_CORREL\_ID ou les deux sont spécifiés en tant qu'options, les valeurs MsgId et CorrelId renvoyées sont également renseignées. (MQPMO\_NEW\_MSG\_ID peut être implicitement spécifié en spécifiant une zone MsgId vide).

Seules les zones indiquées ci-dessus sont renseignées. D'autres informations normalement renvoyées dans la structure MQMD ou MQPMO ne sont pas définies.

Lors de la demande de réponse d'insertion asynchrone pour MQPUT1, les noms ResolvedQName et ResolvedQMgrrenvoyés dans la structure MQOD ne sont pas définis.

Lorsque vous demandez une réponse d'insertion asynchrone pour MQPUT ou MQPUT1, un CompCode et une raison de MQCC\_OK et MQRC\_NONE ne signifient pas nécessairement que le message a été correctement inséré dans une file d'attente. Lors du développement d'une application MQI qui utilise une réponse d'insertion asynchrone et qui requiert la confirmation que des messages ont été insérés dans une file d'attente, vous devez vérifier à la fois les codes CompCode et les codes anomalie des opérations d'insertion et utiliser MQSTAT pour obtenir des informations sur les erreurs asynchrones.

Bien que la réussite ou l'échec de chaque appel MQPUT ou MQPUT1 individuel ne soit pas immédiatement renvoyé, la première erreur qui s'est produite lors d'un appel asynchrone peut être déterminée ultérieurement via un appel à MQSTAT.

Si un message persistant sous le point de synchronisation ne parvient pas à être distribué à l'aide d'une réponse d'insertion asynchrone et que vous tentez de valider la transaction, la validation échoue et la transaction est annulée avec le code achèvement MQCC\_FAILED et le motif MQRC\_BACKED\_OUT. L'application peut appeler MQSTAT pour déterminer la cause d'un échec MQPUT ou MQPUT1 précédent.

### **MQPMO\_SYNC\_REPONSE**

La spécification de ce type de réponse d'insertion garantit que l'opération MQPUT ou MQPUT1 est toujours émise de manière synchrone. Si l'opération d'insertion aboutit, toutes les zones de MQMD et MQPMO sont terminées.

Cette option garantit une réponse synchrone quelle que soit la valeur de réponse d'insertion par défaut définie dans la file d'attente ou l'objet de rubrique.

### **MQPMO\_REPONSE\_AS\_Q\_DEF**

Si cette valeur est spécifiée pour un appel MQPUT, le type de réponse d'insertion utilisé est pris de la valeur DEFPRESP spécifiée dans la file d'attente lors de sa première ouverture par l'application.

- Si la file d'attente est une file d'attente de cluster et que cette valeur est spécifiée pour un appel MQPUT, le type de réponse d'insertion utilisé est issu de l'attribut **DEFPRESP** défini sur le gestionnaire de files d'attente *de destination* qui possède l'instance particulière de la file d'attente dans laquelle le message est placé.

Lorsqu'il existe plusieurs instances de la file d'attente de cluster et que cet attribut est différent pour chaque instance, la valeur définie dans l'une des instances est utilisée, mais il n'est pas possible de prévoir l'instance qui sera choisie. Par conséquent, associez cet attribut à la même valeur dans toutes les instances. Sinon, le message d'erreur AMQ9407 est émis dans les journaux du gestionnaire de files d'attente. Voir aussi [How are destination object attributes resolved for aliases, remote and cluster queues?](#)

- Si la file d'attente n'est pas une file d'attente de cluster et que cette valeur est spécifiée pour un appel MQPUT, le type de réponse d'insertion utilisé est issu de l'attribut **DEFPRESP** défini sur le gestionnaire de files d'attente *local*, même si le gestionnaire de files d'attente de destination est distant.

Si une application client est connectée à un gestionnaire de files d'attente à un niveau antérieur à IBM WebSphere MQ 7.0, elle se comporte comme si MQPMO\_SYNC\_RESPONSE était spécifié.

Si cette option est spécifiée pour un appel MQPUT1, la valeur de l'attribut DEFPRESP n'est pas connue avant l'envoi de la demande au serveur. Par défaut, si l'appel MQPUT1 utilise MQPMO\_SYNCPOINT, il se comporte comme pour MQPMO\_ASYNC\_RESPONSE, et s'il utilise MQPMO\_NO\_SYNCPOINT, il se comporte comme pour MQPMO\_SYNC\_RESPONSE. Toutefois, vous pouvez remplacer ce comportement par défaut en définissant la propriété Put1DefaultAlwaysSync dans le fichier de configuration du client. Voir [Strophe CHANNELS du fichier de configuration du client](#).

### **MQPMO\_REPONSE\_AS\_TOPIC\_DEF**

MQPMO\_RESPONSE\_AS\_TOPIC\_DEF est un synonyme de MQPMO\_RESPONSE\_AS\_Q\_DEF pour une utilisation avec des objets de rubrique.

**Autres options.** Les options suivantes contrôlent la vérification des autorisations, ce qui se produit lorsque le gestionnaire de files d'attente est mis au repos et la résolution des noms de file d'attente et de gestionnaire de files d'attente:

### **MQPMO\_ALTERNATE\_USER\_AUTHORITY**

MQPMO\_ALTERNATE\_USER\_AUTHORITY indique que la zone *AlternateUserId* du paramètre **ObjDesc** de l'appel MQPUT1 contient un identificateur utilisateur qui doit être utilisé pour valider les droits d'accès permettant d'insérer des messages dans la file d'attente. L'appel ne peut aboutir que si *AlternateUserId* est autorisé à ouvrir la file d'attente avec les options spécifiées, que l'ID utilisateur sous lequel l'application s'exécute soit ou non autorisé à le faire. (Cela ne s'applique pas aux options de contexte spécifiées, qui sont toujours vérifiées par rapport à l'identificateur utilisateur sous lequel l'application s'exécute.)

Cette option est valide uniquement avec l'appel MQPUT1 .

### **MQPMO\_FAIL\_IF QUIESCING**

Cette option force l'échec de l'appel MQPUT ou MQPUT1 si le gestionnaire de files d'attente est à l'état de mise au repos.

Sous z/OS, cette option force également l'appel MQPUT ou MQPUT1 à échouer si la connexion (pour une application CICS ou IMS ) est à l'état de mise au repos.

L'appel renvoie le code achèvement MQCC\_FAILED avec le code anomalie MQRC\_Q\_MGR QUIESCING ou MQRC\_CONNECTION QUIESCING.

### **MQPMO\_RESOLVE\_LOCAL\_Q**

Cette option permet de renseigner *ResolvedQName* dans la structure MQPMO avec le nom de la file d'attente locale dans laquelle le message est inséré et *ResolvedQMgrName* avec le nom du gestionnaire de files d'attente local qui héberge la file d'attente locale. Pour plus d'informations sur MQPMO\_RESOLVE\_LOCAL\_Q, voir la rubrique [MQOO\\_RESOLVE\\_LOCAL\\_Q](#).

Si vous êtes autorisé à placer dans une file d'attente, vous disposez des droits requis pour spécifier cet indicateur dans l'appel MQPUT ; aucun droit spécial n'est requis.

**Option par défaut.** Si vous n'avez besoin d'aucune des options décrites, utilisez l'option suivante:

### **MQPMO\_NONE**

Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. Toutes les options sont définies sur leur valeur par défaut. MQPMO\_NONE est défini pour faciliter la documentation du programme ; il n'est pas prévu que cette option soit utilisée avec d'autres, mais comme sa valeur est zéro, une telle utilisation ne peut pas être détectée.

MQPMO\_NONE est une zone d'entrée. La valeur initiale de la zone *Options* est MQPMO\_NONE.

### **Délai d'attente (MQLONG) pour MQPMO**

Il s'agit d'une zone réservée ; sa valeur n'est pas significative. La valeur initiale de cette zone est -1.

### **Contexte (MQHOBJ) pour MQPMO**

Si MQPMO\_PASS\_IDENTITY\_CONTEXT ou MQPMO\_PASS\_ALL\_CONTEXT est spécifié, cette zone doit contenir le descripteur de file d'attente d'entrée à partir duquel les informations de contexte à associer au message en cours d'insertion sont extraites.

Si ni MQPMO\_PASS\_IDENTITY\_CONTEXT ni MQPMO\_PASS\_ALL\_CONTEXT n'est spécifié, cette zone est ignorée.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

### **Nombre de KnownDest(MQLONG) pour MQPMO**

Il s'agit du nombre de messages que l'appel MQPUT ou MQPUT1 a envoyés avec succès aux files d'attente de la liste de distribution qui sont des files d'attente locales. Ce nombre n'inclut pas les messages envoyés à des files d'attente qui se résolvent en files d'attente éloignées (même si une file d'attente de transmission locale est utilisée initialement pour stocker le message). Cette zone est également définie lors de l'insertion d'un message dans une file d'attente unique qui ne figure pas dans une liste de distribution.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0. Cette zone n'est pas définie si *Version* est inférieur à MQPMO\_VERSION\_1.

Cette zone n'est pas définie sur z/OS car les listes de distribution ne sont pas prises en charge.

### **Nombre UnknownDest(MQLONG) pour MQPMO**

Il s'agit du nombre de messages que l'appel MQPUT ou MQPUT1 en cours a correctement envoyés aux files d'attente de la liste de distribution qui sont résolues en files d'attente éloignées. Messages que le gestionnaire de files d'attente conserve temporairement sous forme de liste de distribution, en tant que nombre de destinations individuelles que ces listes de distribution contiennent. Cette zone est également définie lors de l'insertion d'un message dans une file d'attente unique qui ne figure pas dans une liste de distribution.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0. Cette zone n'est pas définie si *Version* est inférieur à MQPMO\_VERSION\_1.

Cette zone n'est pas définie sur z/OS car les listes de distribution ne sont pas prises en charge.

### **InvalidDest(MQLONG) pour MQPMO**

Il s'agit du nombre de messages qui n'ont pas pu être envoyés aux files d'attente de la liste de distribution. Le nombre inclut les files d'attente dont l'ouverture a échoué, ainsi que les files d'attente dont l'ouverture a abouti mais pour lesquelles l'opération d'insertion a échoué. Cette zone est également définie lors de l'insertion d'un message dans une file d'attente unique qui ne figure pas dans une liste de distribution.

**Remarque :** Cette zone est définie si le paramètre **CompCode** de l'appel MQPUT ou MQPUT1 est MQCC\_OK ou MQCC\_WARNING; elle peut être définie si le paramètre **CompCode** est MQCC\_FAILED, mais ne repose pas sur ce paramètre dans le code de l'application.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0. Cette zone n'est pas définie si *Version* est inférieur à MQPMO\_VERSION\_1.

Cette zone n'est pas définie sur z/OS car les listes de distribution ne sont pas prises en charge.

### **ResolvedQName (MQCHAR48) pour MQPMO**

Il s'agit du nom de la file d'attente de destination une fois la résolution de nom effectuée par le gestionnaire de files d'attente local. Le nom renvoyé est le nom d'une file d'attente qui existe sur le gestionnaire de files d'attente identifié par *ResolvedQMgrName*.

Une valeur non vide est renvoyée uniquement si l'objet est une file d'attente unique ; si l'objet est une liste de distribution ou une rubrique, la valeur renvoyée n'est pas définie.

Il s'agit d'une zone de sortie. La longueur de cette zone est indiquée par MQ\_Q\_NAME\_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

### **Nom ResolvedQMgr(MQCHAR48) pour MQPMO**

Il s'agit du nom du gestionnaire de files d'attente de destination une fois la résolution de nom effectuée par le gestionnaire de files d'attente local. Le nom renvoyé correspond au nom du gestionnaire de files d'attente qui possède la file d'attente identifiée par *ResolvedQName* et peut correspondre au nom du gestionnaire de files d'attente local.

Si *ResolvedQName* est une file d'attente partagée appartenant au groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local, *ResolvedQMgrName* est le nom du groupe de partage de files d'attente. Si la file d'attente appartient à un autre groupe de partage de files d'attente, *ResolvedQName* peut être le nom du groupe de partage de files d'attente ou le nom d'un gestionnaire de files d'attente membre du groupe de partage de files d'attente (la nature de la valeur renvoyée est déterminée par les définitions de files d'attente qui existent sur le gestionnaire de files d'attente local).

Une valeur non vide est renvoyée uniquement si l'objet est une file d'attente unique ; si l'objet est une liste de distribution ou une rubrique, la valeur renvoyée n'est pas définie.

Il s'agit d'une zone de sortie. La longueur de cette zone est indiquée par `MQ_Q_MGR_NAME_LENGTH`. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

### ***RecsPresent (MQLONG) pour MQPMO***

Il s'agit du nombre d'enregistrements de message d'insertion MQPMR ou d'enregistrements de réponse MQRR fournis par l'application. Ce nombre peut être supérieur à zéro uniquement si le message est inséré dans une liste de distribution. Les enregistrements de message d'insertion et les enregistrements de réponse sont facultatifs ; l'application n'a pas besoin de fournir d'enregistrements ou elle peut choisir de fournir des enregistrements d'un seul type. Toutefois, si l'application fournit des enregistrements des deux types, elle doit fournir des enregistrements *RecsPresent* de chaque type.

Il n'est pas nécessaire que la valeur de *RecsPresent* soit identique au nombre de destinations dans la liste de distribution. Si un trop grand nombre d'enregistrements est fourni, l'excès n'est pas utilisé ; si le nombre d'enregistrements est trop faible, les valeurs par défaut sont utilisées pour les propriétés de message des destinations qui n'ont pas d'enregistrements de message insérés (voir *PutMsgRecOffset*).

Si *RecsPresent* est inférieur à zéro ou supérieur à zéro mais que le message n'est pas inséré dans une liste de distribution, l'appel échoue avec le code anomalie `MQRC_RECS_Présentat_ERROR`.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à `MQPMO_VERSION_2`.

### ***PutMsgRecFields (MQLONG) pour MQPMO***

Cette zone contient des indicateurs qui indiquent les zones MQPMR présentes dans les enregistrements de message d'insertion fournis par l'application. Utilisez *PutMsgRecFields* uniquement lorsque le message est inséré dans une liste de distribution. La zone est ignorée si *RecsPresent* est égal à zéro, ou si les deux valeurs *PutMsgRecOffset* et *PutMsgRecPtr* sont égales à zéro.

Pour les zones qui sont présentes, le gestionnaire de files d'attente utilise pour chaque destination les valeurs des zones de l'enregistrement de message d'insertion correspondant. Pour les zones absentes, le gestionnaire de files d'attente utilise les valeurs de la structure MQMD.

Utilisez un ou plusieurs des indicateurs suivants pour indiquer les zones qui sont présentes dans les enregistrements de message d'insertion:

#### **ID MQPMRF\_MSG\_ID**

La zone d'identificateur de message est présente.

#### **ID\_CORREL\_MQPMRF\_**

La zone d'identificateur de corrélation est présente.

#### **ID\_GROUPE\_MQPMRF\_ID**

La zone d'identificateur de groupe est présente.

#### **MQPMRF\_FEEDBACK**

Un champ de commentaires est présent.

#### **MQPMRF\_COMPTING\_TOKEN**

La zone de jeton de comptabilité est présente.

Si vous spécifiez cet indicateur, indiquez `MQPMO_SET_IDENTITY_CONTEXT` ou `MQPMO_SET_ALL_CONTEXT` dans la zone *Options* ; si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie `MQRC_PMO_RECORD_FLAGS_ERROR`.

Si aucune zone MQPMR n'est présente, les éléments suivants peuvent être spécifiés:

#### **MQPMRF\_AUCUN**

Aucune zone d'enregistrement d'insertion de message n'est présente.

Si cette valeur est spécifiée, soit *RecsPresent* doit être égal à zéro, soit les deux valeurs *PutMsgRecOffset* et *PutMsgRecPtr* doivent être égales à zéro.



MQPMRF\_NONE est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette constante soit utilisée avec d'autres, mais comme sa valeur est nulle, une telle utilisation ne peut pas être détectée.

Si *PutMsgRecFields* contient des indicateurs non valides ou que des enregistrements de message d'insertion sont fournis mais que *PutMsgRecFields* a la valeur MQPMRF\_NONE, l'appel échoue avec le code anomalie MQRC\_PMO\_RECORD\_FLAGS\_ERROR.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est MQPMRF\_NONE. Cette zone est ignorée si *Version* est inférieur à MQPMO\_VERSION\_2.

### ***PutMsgRecOffset (MQLONG) pour MQPMO***

Décalage en octets du premier enregistrement de message d'insertion MQPMR à partir du début de la structure MQPMO. Le décalage peut être positif ou négatif. *PutMsgRecOffset* est utilisé uniquement lorsque le message est inséré dans une liste de distribution. La zone est ignorée si *RecsPresent* a la valeur zéro.

Lorsque le message est inséré dans une liste de distribution, un tableau d'un ou de plusieurs enregistrements de message d'insertion MQPMR peut être fourni afin de spécifier certaines propriétés du message pour chaque destination individuellement ; ces propriétés sont les suivantes :

- Identificateur de message
- Identificateur de corrélation
- Identificateur de groupe
- Valeur de retour d'informations
- Jeton de comptabilité

Vous n'avez pas besoin de spécifier toutes ces propriétés, mais quel que soit le sous-ensemble que vous choisirez, spécifiez les zones dans le bon ordre. Pour plus de détails, voir la description de la structure MQPMR.

Généralement, il doit y avoir autant d'enregistrements de message d'insertion que d'enregistrements d'objet spécifiés par MQOD lors de l'ouverture de la liste de distribution ; chaque enregistrement de message d'insertion fournit les propriétés de message pour la file d'attente identifiée par l'enregistrement d'objet correspondant. Les files d'attente de la liste de distribution dont l'ouverture a échoué doivent toujours avoir des enregistrements de message alloués aux emplacements appropriés dans le tableau, bien que les propriétés de message soient ignorées dans ce cas.

Le nombre d'enregistrements de message d'insertion peut être différent du nombre d'enregistrements d'objet. S'il y a moins d'enregistrements de message d'insertion que d'enregistrements d'objet, les propriétés de message pour les destinations qui n'ont pas d'enregistrements de message d'insertion sont extraites des zones correspondantes dans le descripteur de message MQMD. S'il y a plus d'enregistrements de message d'insertion que d'enregistrements d'objet, l'excès n'est pas utilisé (bien qu'il soit toujours possible d'y accéder). Les enregistrements de message d'insertion sont facultatifs, mais s'ils sont fournis, ils doivent être *RecsPresent* .

Fournissez les enregistrements de message d'insertion de la même manière que les enregistrements d'objet dans MQOD, soit en spécifiant un décalage dans *PutMsgRecOffset*, soit en spécifiant une adresse dans *PutMsgRecPtr* ; Pour plus d'informations sur cette opération, voir la zone *ObjectRecOffset* décrite dans «MQOD-Descripteur d'objet», à la page 501.

Vous ne pouvez pas utiliser plus d'un *PutMsgRecOffset* ou *PutMsgRecPtr* ; l'appel échoue avec le code anomalie MQRC\_PUT\_MSG\_RECORDS\_ERROR si les deux sont différents de zéro.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQPMO\_VERSION\_2.

### ***ResponseRecDécalage (MQLONG) pour MQPMO***

Décalage en octets du premier enregistrement de réponse MQRR à partir du début de la structure MQPMO. Le décalage peut être positif ou négatif. *ResponseRecOffset* est utilisé uniquement lorsque le message est inséré dans une liste de distribution. La zone est ignorée si *RecsPresent* a la valeur zéro.

Lors de l'insertion du message dans une liste de distribution, vous pouvez fournir un tableau d'un ou de plusieurs enregistrements de réponse MQRR afin d'identifier les files d'attente auxquelles le message n'a pas été envoyé avec succès (*zoneCompCode* dans MQRR), ainsi que la raison de chaque échec (*zoneReason* dans MQRR). Il se peut que le message n'ait pas été envoyé en raison de l'échec de l'ouverture de la file d'attente ou de l'échec de l'opération d'insertion. Le gestionnaire de files d'attente définit les enregistrements de réponse uniquement lorsque le résultat de l'appel est mixte (c'est-à-dire que certains messages ont été envoyés avec succès alors que d'autres ont échoué, ou que tous ont échoué, mais pour des raisons différentes) ; le code anomalie MQRC\_MULTIPLE\_MOTIFS de l'appel indique ce cas. Si le même code anomalie s'applique à toutes les files d'attente, ce code anomalie est renvoyé dans le paramètre **Reason** de l'appel MQPUT ou MQPUT1 et les enregistrements de réponse ne sont pas définis.

En général, il existe autant d'enregistrements de réponse que d'enregistrements d'objet spécifiés par MQOD lors de l'ouverture de la liste de distribution ; si nécessaire, chaque enregistrement de réponse est défini sur le code achèvement et le code raison de l'insertion dans la file d'attente identifiée par l'enregistrement d'objet correspondant. Les files d'attente de la liste de distribution dont l'ouverture a échoué doivent toujours avoir des enregistrements de réponse alloués aux emplacements appropriés de la grappe, bien qu'elles soient définies sur le code achèvement et le code raison résultant de l'opération d'ouverture, plutôt que sur l'opération d'insertion.

Le nombre d'enregistrements de réponse peut différer du nombre d'enregistrements d'objet. S'il y a moins d'enregistrements de réponse que d'enregistrements d'objet, l'application peut ne pas être en mesure d'identifier toutes les destinations pour lesquelles l'opération d'insertion a échoué ou les raisons des échecs. S'il y a plus d'enregistrements de réponse que d'enregistrements d'objet, l'excédent n'est pas utilisé (bien qu'il soit toujours possible d'y accéder). Les enregistrements de réponse sont facultatifs, mais s'ils sont fournis, ils doivent être *RecsPresent*.

Fournissez les enregistrements de réponse de la même manière que les enregistrements d'objet dans MQOD, soit en spécifiant un décalage dans *ResponseRecOffset*, soit en spécifiant une adresse dans *ResponseRecPtr* ; Pour plus d'informations sur cette opération, voir la zone *ObjectRecOffset* décrite dans «MQOD-Descripteur d'objet», à la page 501. Toutefois, n'utilisez pas plus de *ResponseRecOffset* et *ResponseRecPtr* ; l'appel échoue avec le code anomalie MQRC\_RESPONSE\_RECORDS\_ERROR si les deux sont différents de zéro.

Pour l'appel MQPUT1, cette zone doit être à zéro. En effet, les informations de réponse (si elles sont demandées) sont renvoyées dans les enregistrements de réponse spécifiés par le descripteur d'objet MQOD.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQPMO\_VERSION\_2.

### ***PutMsgRecPtr (MQPTR) pour MQPMO***

Il s'agit de l'adresse du premier enregistrement de message d'insertion MQPMR. Utilisez *PutMsgRecPtr* uniquement lorsque le message est inséré dans une liste de distribution. La zone est ignorée si *RecsPresent* a la valeur zéro.

Vous pouvez utiliser *PutMsgRecPtr* ou *PutMsgRecOffset* pour spécifier les enregistrements de message d'insertion, mais pas les deux. Pour plus de détails, voir «*PutMsgRecOffset (MQLONG) pour MQPMO*», à la page 537. Si vous n'utilisez pas *PutMsgRecPtr*, définissez-le sur le pointeur null ou sur les octets null.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire. Cette zone est ignorée si *Version* est inférieur à MQPMO\_VERSION\_2.

**Remarque :** Sur les plateformes où le langage de programmation ne prend pas en charge le type de données pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée, la valeur initiale étant la chaîne d'octets tout-null.

### ***ResponseRecPtr (MQPTR) pour MQPMO***

Il s'agit de l'adresse du premier enregistrement de réponse MQRR. *ResponseRecPtr* est utilisé uniquement lorsque le message est inséré dans une liste de distribution. La zone est ignorée si *RecsPresent* a la valeur zéro.

Utilisez *ResponseRecPtr* ou *ResponseRecOffset* pour spécifier les enregistrements de réponse, mais pas les deux. Pour plus de détails, voir «*ResponseRecDécalage (MQLONG) pour MQPMO*», à la page 537. Si vous n'utilisez pas *ResponseRecPtr*, définissez-le sur le pointeur null ou sur les octets null.

Pour l'appel MQPUT1, cette zone doit être le pointeur null ou des octets null. En effet, les informations de réponse (si elles sont demandées) sont renvoyées dans les enregistrements de réponse spécifiés par le descripteur d'objet MQOD.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire. Cette zone est ignorée si *Version* est inférieur à MQPMO\_VERSION\_2.

**Remarque :** Sur les plateformes où le langage de programmation ne prend pas en charge le type de données pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée, la valeur initiale étant la chaîne d'octets tout-null.

### ***Descripteur OriginalMsg(MQHMSG) pour MQPMO***

Il s'agit d'un descripteur facultatif d'un message. Il se peut qu'il ait été précédemment extrait d'une file d'attente. L'utilisation de ce descripteur est soumise à la valeur de la zone *Action*; voir aussi [NewMsgHandle](#).

Le contenu du descripteur de message d'origine ne sera pas modifié par l'appel MQPUT ou MQPUT1.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est MQHM\_NONE. Cette zone est ignorée si la version est inférieure à MQPMO\_VERSION\_3.

### ***NewMsgIdentificateur (MQHMSG) pour MQPMO***

Il s'agit d'un descripteur facultatif pour le message en cours d'insertion en fonction de la valeur de la zone *Action*. Il définit les propriétés du message et remplace les valeurs de *OriginalMsgHandle*, si elles sont spécifiées.

En cas de retour de l'appel MQPUT ou MQPUT1, le contenu du descripteur reflète le message qui a été inséré.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est MQHM\_NONE. Cette zone est ignorée si la version est inférieure à MQPMO\_VERSION\_3.

### ***Action (MQLONG) pour MQPMO***

Indique le type d'insertion effectuée et la relation entre le message d'origine spécifié par la zone Descripteur OriginalMsg et le nouveau message spécifié par la zone Descripteur NewMsg. Les propriétés du message sont choisies par le gestionnaire de files d'attente en fonction de la valeur de l'action spécifiée.

Vous pouvez choisir de fournir le contenu du descripteur de message à l'aide du paramètre *MsgDesc* dans les appels MQPUT ou MQPUT1. Il est également possible de ne pas fournir le paramètre *MsgDesc* ou de spécifier qu'il s'agit d'une sortie uniquement en incluant MQPMO\_MD\_FOR\_OUTPUT\_ONLY dans la zone Options de la structure MQPMO.

Si le paramètre MsgDesc n'est pas fourni ou s'il est spécifié comme étant uniquement en sortie, le descripteur de message du nouveau message est renseigné à partir des zones de descripteur de message du MQPMO, conformément aux règles décrites dans cette rubrique.

Le paramètre de contexte et les activités de transmission décrites dans Contrôle des informations de contexte prennent effet une fois que le descripteur de message a été composé.

Si une valeur d'action incorrecte est spécifiée, l'appel échoue avec le code anomalie MQRC\_ACTION\_ERROR.

L'une des actions suivantes peut être spécifiée:

#### **MQACTP\_NOUVEAU**

Un nouveau message est inséré et aucune relation avec un message précédent n'est indiquée par le programme. Le descripteur de message se compose comme suit:

- Si une valeur MsgDesc est fournie dans l'appel MQPUT ou MQPUT1 et que MQPMO\_MD\_FOR\_OUTPUT\_ONLY ne figure pas dans MQPMO.Options, utilisé comme descripteur de message non modifié.
- Si MsgDesc n'est pas fourni, ou si MQPMO\_MD\_FOR\_OUTPUT\_ONLY se trouve dans MQPMO.Options : le gestionnaire de files d'attente génère le descripteur de message à l'aide d'une combinaison de propriétés des identificateurs OriginalMsg et NewMsg. Les zones de descripteur de message définies explicitement sur le nouveau descripteur de message sont prioritaires par rapport à celles du descripteur de message d'origine.

Les données de message sont extraites du paramètre de mémoire tampon MQPUT ou MQPUT1 .

#### **MQACTP\_FORWARD**

Un message précédemment extrait est en cours de transfert. Le descripteur de message d'origine indique le message précédemment extrait.

Le nouveau descripteur de message spécifie toutes les modifications apportées aux propriétés (y compris celles du descripteur de message) dans le descripteur de message d'origine.

Le descripteur de message se compose comme suit:

- Si une valeur MsgDesc est fournie dans l'appel MQPUT ou MQPUT1 et que MQPMO\_MD\_FOR\_OUTPUT\_ONLY ne figure pas dans MQPMO.Options, utilisé comme descripteur de message non modifié.
- Si MsgDesc n'est pas fourni, ou si MQPMO\_MD\_FOR\_OUTPUT\_ONLY se trouve dans MQPMO.Options : le gestionnaire de files d'attente génère le descripteur de message à l'aide d'une combinaison de propriétés des identificateurs OriginalMsg et NewMsg. Les zones de descripteur de message définies explicitement sur le nouveau descripteur de message sont prioritaires par rapport à celles du descripteur de message d'origine.
- Si MQPMO\_NEW\_MSG\_ID ou MQPMO\_NEW\_CORREL\_ID sont spécifiés dans MQPMO.Options sont alors honorées.

Les propriétés de message sont composées comme suit:

- Toutes les propriétés du descripteur de message d'origine qui ont MQCOPY\_FORWARD dans MQPD.CopyOptions
- Toutes les propriétés du nouveau descripteur de message. Pour chaque propriété du nouveau descripteur de message ayant le même nom qu'une propriété du descripteur de message d'origine, la valeur est extraite du nouveau descripteur de message. La seule exception à cette règle est le cas particulier où la propriété du nouveau descripteur de message porte le même nom qu'une propriété du descripteur de message d'origine, mais que la valeur de la propriété est null. Dans ce cas, la propriété est supprimée du message.

Les données de message à transmettre sont extraites du paramètre de mémoire tampon MQPUT ou MQPUT1 .

## MQACTP\_REPLY

Une réponse est en cours à un message précédemment extrait. Le descripteur de message d'origine indique le message précédemment extrait.

Le nouveau descripteur de message spécifie toutes les modifications apportées aux propriétés (y compris celles du descripteur de message) dans le descripteur de message d'origine.

Le descripteur de message se compose comme suit:

- Si une valeur MsgDesc est fournie dans l'appel MQPUT ou MQPUT1 et que MQPMO\_MD\_FOR\_OUTPUT\_ONLY ne figure pas dans MQPMO.Options, utilisé comme descripteur de message non modifié.
- Si MsgDesc n'est pas fourni, ou si MQPMO\_MD\_FOR\_OUTPUT\_ONLY se trouve dans MQPMO.Options, puis les zones de descripteur de message initial sont choisies comme suit:

Zone dans MQMD	Valeur utilisée
Rapport	Si MQRO_PASS_DISCARD_AND_EXPIRATION et MQRO_DISCARD_MSG sont définis: MQRO_DISCARD_MSG sinon MQRO_AUCUN
MsgType	MQMT_REPLY
Expiration	Si MQRO_PASS_DISCARD_AND_EXPIRATION est défini: Copié à partir du message d'entrée sinon MQEI_UNLIMITED
Commentaires	MQFB_AUCUN
MsgId	Si MQPMO_NEW_MSG_ID est défini: Un nouvel identificateur de message est généré else if MQRO_PASS_MSG_ID est défini: Copié à partir du message d'entrée sinon MQMI_AUCUN
CorrelId	Si MQPMO_NEW_CORREL_ID est défini: Un nouvel identificateur de corrélation est généré else if MQRO_COPY_MSG_ID_TO_CORREL_ID est défini: Copié à partir de la zone MsgId de la message d'entrée sinon, si MQRO_PASS_CORREL_ID est défini: Copié à partir de la zone CorrelId de la message d'entrée sinon MQCI_NONE
BackoutCount	0
ReplyToQ	Espaces vides
ReplyToQMgr	Espaces vides

Tableau 509. Transformation du descripteur de message de réponse (suite)

Zone dans MQMD	Valeur utilisée
GroupId	MQGI_AUCUN
MsgSeqNumber	1
Décalage	0
MsgFlags	MQMF_AUCUN
OriginalLength	MQOL_UNDEFINI

- Le descripteur de message est ensuite modifié par le nouveau descripteur de message-toutes les zones de descripteur de message explicitement définies en tant que propriétés dans le nouveau descripteur de message sont prioritaires sur les zones de descripteur de message comme décrit précédemment.

Les propriétés de message sont composées comme suit:

- Toutes les propriétés du descripteur de message d'origine qui ont MQCOPY\_REPLY dans MQPD.CopyOptions
- Toutes les propriétés du nouveau descripteur de message. Pour chaque propriété du nouveau descripteur de message ayant le même nom qu'une propriété du descripteur de message d'origine, la valeur est extraite du nouveau descripteur de message. La seule exception à cette règle est le cas particulier où la propriété du nouveau descripteur de message porte le même nom qu'une propriété du descripteur de message d'origine, mais que la valeur de la propriété est null. Dans ce cas, la propriété est supprimée du message.

Les données de message à transmettre sont extraites du paramètre de mémoire tampon MQPUT/MQPUT1 .

#### RAPPORT MQACTP\_RAPPORT

Un rapport est généré suite à un message précédemment extrait. Le descripteur de message d'origine indique le message à l'origine de la génération du rapport.

Le nouveau descripteur de message spécifie toutes les modifications apportées aux propriétés (y compris celles du descripteur de message) dans le descripteur de message d'origine.

Le descripteur de message se compose comme suit:

- Si une valeur MsgDesc est fournie dans l'appel MQPUT ou MQPUT1 et que MQPMO\_MD\_FOR\_OUTPUT\_ONLY ne figure pas dans MQPMO.Options, utilisé comme descripteur de message non modifié.
- Si MsgDesc n'est pas fourni, ou si MQPMO\_MD\_FOR\_OUTPUT\_ONLY se trouve dans MQPMO.Options et les zones de descripteur de message initial sont choisies comme suit:

Tableau 510. Transformation du descripteur de message de rapport

Zone dans MQMD	Valeur utilisée
Rapport	Si MQRO_PASS_DISCARD_AND_EXPIRATION et MQRO_DISCARD_MSG sont définis: MQRO_DISCARD_MSG sinon MQRO_AUCUN
MsgType	MQMT_REPORT

Tableau 510. Transformation du descripteur de message de rapport (suite)

Zone dans MQMD	Valeur utilisée
Expiration	Si MQRO_PASS_DISCARD_AND_EXPIRATION est défini: Copié à partir du message d'entrée sinon MQEI_UNLIMITED
MsgId	Si MQPMO_NEW_MSG_ID est défini: Un nouvel identificateur de message est généré else if MQRO_PASS_MSG_ID est défini: Copié à partir du message d'entrée sinon MQMI_AUCUN
CorrelId	Si MQPMO_NEW_CORREL_ID est défini: Un nouvel identificateur de corrélation est généré else if MQRO_COPY_MSG_ID_TO_CORREL_ID est défini: Copié à partir de la zone MsgId de la message d'entrée sinon, si MQRO_PASS_CORREL_ID est défini: Copié à partir de la zone CorrelId de la message d'entrée sinon MQCI_NONE
BackoutCount	0
ReplyToQ	Espaces vides
ReplyToQMgr	Espaces vides
OriginalLength	Défini sur <i>BufferLength</i>

- Le descripteur de message est ensuite modifié par le nouveau descripteur de message-toutes les zones de descripteur de message explicitement définies en tant que propriétés dans le nouveau descripteur de message sont prioritaires sur les zones de descripteur de message comme décrit précédemment.

Les propriétés de message sont composées comme suit:

- Toutes les propriétés du descripteur de message d'origine qui comportent MQCOPY\_REPORT dans MQPD.CopyOptions
- Toutes les propriétés du nouveau descripteur de message. Pour chaque propriété du nouveau descripteur de message ayant le même nom qu'une propriété du descripteur de message d'origine, la valeur est extraite du nouveau descripteur de message. La seule exception à cette règle est le cas particulier où la propriété du nouveau descripteur de message porte le même nom qu'une propriété du descripteur de message d'origine, mais que la valeur de la propriété est null. Dans ce cas, la propriété est supprimée du message.

La zone Feedback dans le MQMD résultant représente le rapport à générer. Une valeur de commentaire en retour de MQFB\_NONE entraîne l'échec de l'appel MQPUT ou MQPUT1 avec le code anomalie MQRC\_FEEDBACK\_ERROR.

Pour choisir les données utilisateur du message de rapport, IBM MQ consulte les zones Rapport et Commentaires dans le MQMD résultant, ainsi que les paramètres Buffer et BufferLength de l'appel MQPUT ou MQPUT1 .

- Si les commentaires en retour sont MQFB\_COA, MQFB\_COD ou MQFB\_EXPIRATION, la valeur du rapport est inspectée.
- Si l'un des cas suivants est vrai, les données de message complètes de Buffer pour une longueur de BufferLength sont utilisées.
  - Les commentaires en retour sont MQFB\_EXPIRATION et le rapport contient MQRO\_EXPIRATION\_WITH\_FULL\_DATA
  - Les commentaires en retour sont MQFB\_COD et le rapport contient MQRO\_COD\_WITH\_FULL\_DATA
  - Les commentaires en retour sont MQFB\_COA et le rapport contient MQRO\_COA\_WITH\_FULL\_DATA
- Si l'un des cas suivants est vrai, les 100 premiers octets du message (ou BufferLength si cette valeur est inférieure à 100) de Buffer sont utilisés.
  - Le retour d'informations est MQFB\_EXPIRATION et le rapport contient MQRO\_EXPIRATION\_WITH\_DATA
  - Les commentaires en retour sont MQFB\_COD et le rapport contient MQRO\_COD\_WITH\_DATA
  - Les commentaires en retour sont MQFB\_COA et le rapport contient MQRO\_COA\_WITH\_DATA
- Si les commentaires en retour sont MQFB\_EXPIRATION, MQFB\_COD ou MQFB\_COA et que le rapport ne contient pas les options \*\_WITH\_FULL\_DATA ou \*\_WITH\_DATA associées à cette valeur de rétrofacturation, aucune donnée utilisateur n'est incluse dans le message.
- Si les commentaires en retour prennent une valeur différente de celles répertoriées ci-dessus, Buffer et BufferLength sont utilisés normalement.

La dérivation des données utilisateur décrites dans la liste précédente est également présentée dans le tableau suivant:

<i>Tableau 511. Source des données utilisateur</i>			
	<b>MQFB_COA</b>	<b>MQFB_COD</b>	<b>EXPIRATION_MQFB</b>
<b>MQRO_EXPIRATION_WITH_FULL_DATA</b>	Aucun	Aucun	Mémoire tampon (longueur de la mémoire tampon)
<b>MQRO_COD_WITH_FULL_DATA</b>	Aucun	Mémoire tampon (longueur de la mémoire tampon)	Aucun
<b>MQRO_COA_WITH_FULL_DATA</b>	Mémoire tampon (longueur de la mémoire tampon)	Aucun	Aucun
<b>MQRO_EXPIRATION_WITH_DATA</b>	Aucun	Aucun	Mémoire tampon (100 premiers octets)
<b>MQRO_COD_WITH_DATA</b>	Aucun	Mémoire tampon (100 premiers octets)	Aucun
<b>MQRO_COA_WITH_DATA</b>	Mémoire tampon (100 premiers octets)	Aucun	Aucun

### **PubLevel (MQLONG) pour MQPMO**

La valeur initiale de cette zone est 9. Niveau d'abonnement ciblé par cette publication. Seuls les abonnements dont le SubLevel le plus élevé est inférieur ou égal à cette valeur reçoivent cette publication. Cette valeur doit être comprise entre zéro et 9 ; zéro est le niveau le plus bas. Toutefois, si une publication a été conservée, elle n'est plus disponible pour les abonnés de niveau supérieur car elle est republiée sur le site PubLevel 1.

Pour plus d'informations, voir [Intercepting publications](#).



## MQPMR-Enregistrement de message d'insertion

Utilisez la structure MQPMR pour spécifier diverses propriétés de message pour une destination unique lors de l'insertion d'un message dans une liste de distribution. MQPMR est une structure d'entrée/sortie pour les appels MQPUT et MQPUT1 .

### Disponibilité

La structure MQPMR est disponible sur les plateformes suivantes:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour les clients IBM MQ connectés à ces systèmes.

### Jeu de caractères et codage

Les données dans MQPMR doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et le codage du gestionnaire de files d'attente local indiqué par MQENC\_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ , la structure doit être dans le jeu de caractères et le codage du client.

### Utilisation

En fournissant un tableau de ces structures sur l'appel MQPUT ou MQPUT1 , vous pouvez spécifier des valeurs différentes pour chaque file d'attente de destination dans une liste de distribution. Certaines des zones sont des entrées uniquement, d'autres sont des entrées / sorties.

**Remarque :** Cette structure est inhabituelle dans la mesure où elle n'a pas de présentation fixe. Les zones de cette structure sont facultatives et la présence ou l'absence de chaque zone est indiquée par les indicateurs de la zone *PutMsgRecFields* dans MQPMO. Les zones présentes dans **doivent apparaître dans l'ordre suivant :**

- *MsgId*
- *CorrelId*
- *GroupId*
- *Feedback*
- *AccountingToken*

Les zones absentes n'occupent aucun espace dans l'enregistrement.

Comme MQPMR n'a pas de présentation fixe, aucune définition de celle-ci n'est fournie dans les fichiers d'en-tête, COPY et INCLUDE pour les langages de programmation pris en charge. Le programmeur d'application doit créer une déclaration contenant les zones requises par l'application et définir les indicateurs dans *PutMsgRecFields* pour indiquer les zones présentes.

### Zones

Aucune valeur initiale n'est définie pour cette structure, car aucune déclaration de structure n'est fournie dans les fichiers d'en-tête, COPY et INCLUDE pour les langages de programmation pris en charge. Les exemples de déclaration montrent comment déclarer la structure si toutes les zones sont requises.

Tableau 512. Zones dans MQPMR

Nom de zone	Description de la zone
<u>MsgId</u>	Identificateur de message
<u>CorrelId</u>	Identificateur de corrélation
<u>GroupId</u>	Identificateur de groupe
<u>Commentaires</u>	Commentaires en retour ou code anomalie
<u>AccountingToken</u>	Jeton de comptabilité

## Déclarations de langue

### Déclaration C pour MQPMR

```
typedef struct tagMQPMR MQPMR;
struct tagMQPMR {
    MQBYTE24  MsgId;           /* Message identifier */
    MQBYTE24  CorrelId;       /* Correlation identifier */
    MQBYTE24  GroupId;       /* Group identifier */
    MQLONG    Feedback;      /* Feedback or reason code */
    MQBYTE32  AccountingToken; /* Accounting token */
};
```

### Déclaration COBOL pour MQPMR

```
** MQPMR structure
10 MQPMR.
** Message identifier
15 MQPMR-MSGID PIC X(24).
** Correlation identifier
15 MQPMR-CORRELID PIC X(24).
** Group identifier
15 MQPMR-GROUPID PIC X(24).
** Feedback or reason code
15 MQPMR-FEEDBACK PIC S9(9) BINARY.
** Accounting token
15 MQPMR-ACCOUNTINGTOKEN PIC X(32).
```

### Déclaration PL/I pour MQPMR

```
dcl
1 MQPMR based,
3 MsgId char(24), /* Message identifier */
3 CorrelId char(24), /* Correlation identifier */
3 GroupId char(24), /* Group identifier */
3 Feedback fixed bin(31), /* Feedback or reason code */
3 AccountingToken char(32); /* Accounting token */
```

### Déclaration Visual Basic pour MQPMR

```
Type MQPMR
MsgId As MQBYTE24 'Message identifier'
CorrelId As MQBYTE24 'Correlation identifier'
GroupId As MQBYTE24 'Group identifier'
Feedback As Long 'Feedback or reason code'
AccountingToken As MQBYTE32 'Accounting token'
End Type
```

## **MsgId (MQBYTE24) pour MQPMR**

Il s'agit de l'identificateur de message à utiliser pour le message envoyé à la file d'attente avec un nom qui a été spécifié par l'élément correspondant dans le tableau de structures MQOR fourni sur l'appel

MQOPEN ou MQPUT1 . Il est traité de la même manière que la zone *MsgId* dans MQMD pour une insertion dans une file d'attente unique.

Si cette zone n'est pas présente dans l'enregistrement MQPMR ou s'il y a moins d'enregistrements MQPMR que de destinations, la valeur de MQMD est utilisée pour les destinations qui n'ont pas d'enregistrement MQPMR contenant une zone *MsgId* . Si cette valeur est MQMI\_NONE, un nouvel identificateur de message est généré pour *chacune* de ces destinations (c'est-à-dire que deux de ces destinations n'ont pas le même identificateur de message).

Si MQPMO\_NEW\_MSG\_ID est spécifié, de nouveaux identificateurs de message sont générés pour toutes les destinations de la liste de distribution, qu'elles comportent ou non des enregistrements MQPMR. Cette méthode est différente de celle utilisée pour le traitement de MQPMO\_NEW\_CORREL\_ID (voir la zone *CorrelId*).

Il s'agit d'une zone d'entrée/sortie.

### ***CorrelId (MQBYTE24) pour MQPMR***

Il s'agit de l'identificateur de corrélation à utiliser pour le message envoyé à la file d'attente avec un nom qui a été spécifié par l'élément correspondant dans le tableau de structures MQOR fourni sur l'appel MQOPEN ou MQPUT1 . Il est traité de la même manière que la zone *CorrelId* dans MQMD pour une insertion dans une file d'attente unique.

Si cette zone n'est pas présente dans l'enregistrement MQPMR ou s'il y a moins d'enregistrements MQPMR que de destinations, la valeur de MQMD est utilisée pour les destinations qui n'ont pas d'enregistrement MQPMR contenant une zone *CorrelId* .

Si MQPMO\_NEW\_CORREL\_ID est spécifié, un nouvel identificateur de corrélation *unique* est généré et utilisé pour toutes les destinations de la liste de distribution, qu'elles comportent ou non des enregistrements MQPMR. Cette méthode est différente de celle utilisée pour le traitement de MQPMO\_NEW\_MSG\_ID (voir la zone *MsgId*).

Il s'agit d'une zone d'entrée/sortie.

### ***GroupId (MQBYTE24) pour MQPMR***

GroupId est l'identificateur de groupe à utiliser pour le message envoyé à la file d'attente avec le nom spécifié par l'élément correspondant dans le tableau de structures MQOR fourni dans l'appel MQOPEN ou MQPUT1 . Il est traité de la même manière que la zone *GroupId* dans MQMD pour une insertion dans une file d'attente unique.

Si cette zone n'est pas présente dans l'enregistrement MQPMR ou s'il y a moins d'enregistrements MQPMR que de destinations, la valeur de MQMD est utilisée pour les destinations qui n'ont pas d'enregistrement MQPMR contenant une zone *GroupId* . La valeur est traitée comme indiqué dans Ordre physique dans une file d'attente, mais avec les différences suivantes:

- GroupId est créé à partir du nom de gestionnaire de files d'attente et d'un horodatage. Par conséquent, pour conserver un ID de groupe ( *GroupId* ) unique, les noms de gestionnaire de files d'attente doivent également être uniques. Ne redéfinissez pas non plus les horloges sur la machine des gestionnaires de files d'attente.
- Dans les cas où un nouvel identificateur de groupe serait utilisé, le gestionnaire de files d'attente génère un identificateur de groupe différent pour chaque destination (c'est-à-dire que deux destinations n'ont pas le même identificateur de groupe).
- Dans les cas où la valeur de la zone serait utilisée, l'appel échoue avec le code anomalie MQRC\_GROUP\_ID\_ERROR

Il s'agit d'une zone d'entrée/sortie.

### ***Commentaires en retour (MQLONG) pour MQPMR***

Il s'agit du code retour à utiliser pour le message envoyé à la file d'attente avec le nom qui a été spécifié par l'élément correspondant dans le tableau des structures MQOR fournies sur l'appel MQOPEN

ou MQPUT1 . Il est traité de la même manière que la zone *Feedback* dans MQMD pour une insertion dans une file d'attente unique.

Si cette zone n'est pas présente, la valeur de MQMD est utilisée.

Il s'agit d'une zone d'entrée.

### **AccountingToken (MQBYTE32) pour MQPMR**

Il s'agit du jeton de comptabilité à utiliser pour le message envoyé à la file d'attente avec le nom indiqué par l'élément correspondant dans le tableau des structures MQOR fournies sur l'appel MQOPEN ou MQPUT1 . Il est traité de la même manière que la zone *AccountingToken* dans MQMD pour une insertion dans une file d'attente unique. Voir la description de *AccountingToken* dans «MQMD- Descripteur de message», à la page 438 pour plus d'informations sur le contenu de cette zone.

Si cette zone n'est pas présente, la valeur de MQMD est utilisée.

Il s'agit d'une zone d'entrée.

### **MQRFH-Règles et en-tête de formatage**

La structure MQRFH définit la présentation des règles et l'en-tête de formatage. Utilisez cet en-tête pour envoyer des données de chaîne sous la forme de paires nom-valeur.

### **Disponibilité**

Tous les systèmes IBM MQ , plus IBM MQ MQI clients connectés à ces systèmes.

### **Format**

MQFMT\_RF\_HEADER

### **Jeu de caractères et codage**

Les zones de la structure MQRFH (y compris *NameValueString*) se trouvent dans le jeu de caractères et le codage donnés par les zones *CodedCharSetId* et *Encoding* de la structure d'en-tête qui précède MQRFH, ou par les zones de la structure MQMD si MQRFH se trouve au début des données de message d'application.

Le jeu de caractères doit comporter des caractères mono-octet pour les caractères admis dans les noms de file d'attente.

### **Zones**

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID MQRFH_STRUC_ID	'RFH↵'
<u>Version</u> (numéro de version de la structure)	MQRFH_VERSION_1	1
<u>StrucLength</u> (longueur en octets de la structure MQRFH)	MQRFH_STRUC_LENGT H_FIXED	32
<u>Codage</u> (codage numérique des données qui suivent <i>NameValueString</i> )	MQENC_NATIVE	Dépend de l'environnement

Tableau 513. Zones de MQRFH pour MQRFH (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
CodedCharSetId (indique l'identificateur de jeu de caractères des données qui suivent <i>NameValueString</i> )	MQCCSI_UNDEFINI	0
Format (nom de format des données qui suivent <i>NameValueString</i> )	MQFMT_AUCUN	Espaces vides
Indicateurs (indicateurs)	MQRFH_AUCUN	0
NameValueChaîne (chaîne de caractères de longueur variable contenant des paires nom-valeur)	Aucun	Aucun

**Remarques :**

1. Le symbole ~ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable macro MQRFH\_DEFAULT contient les valeurs répertoriées dans la table. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQRFH MyRFH = {MQRFH_DEFAULT};
```

## Déclarations de langue

### Déclaration C pour MQRFH

```
typedef struct tagMQRFH MQRFH;
struct tagMQRFH {
    MQCHAR4  StrucId;          /* Structure identifier */
    MQLONG   Version;         /* Structure version number */
    MQLONG   StrucLength;     /* Total length of MQRFH including
                             NameValueString */
    MQLONG   Encoding;        /* Numeric encoding of data that follows
                             NameValueString */
    MQLONG   CodedCharSetId; /* Character set identifier of data that
                             follows NameValueString */
    MQCHAR8  Format;          /* Format name of data that follows
                             NameValueString */
    MQLONG   Flags;           /* Flags */
};
```

### Déclaration COBOL pour MQRFH

```
** MQRFH structure
10 MQRFH.
** Structure identifier
15 MQRFH-STRUCID PIC X(4).
** Structure version number
15 MQRFH-VERSION PIC S9(9) BINARY.
** Total length of MQRFH including NAMEVALUESTRING
15 MQRFH-STRUCLNGTH PIC S9(9) BINARY.
** Numeric encoding of data that follows NAMEVALUESTRING
15 MQRFH-ENCODING PIC S9(9) BINARY.
** Character set identifier of data that follows NAMEVALUESTRING
15 MQRFH-CODEDCHARSETID PIC S9(9) BINARY.
** Format name of data that follows NAMEVALUESTRING
15 MQRFH-FORMAT PIC X(8).
** Flags
15 MQRFH-FLAGS PIC S9(9) BINARY.
```

## Déclaration PL/I pour MQRFH

```
dc1
1 MQRFH based,
3 StrucId      char(4),      /* Structure identifier */
3 Version      fixed bin(31), /* Structure version number */
3 StrucLength  fixed bin(31), /* Total length of MQRFH including
                             NameValueString */
3 Encoding     fixed bin(31), /* Numeric encoding of data that
                             follows NameValueString */
3 CodedCharSetId fixed bin(31), /* Character set identifier of data
                             that follows NameValueString */
3 Format        char(8),      /* Format name of data that follows
                             NameValueString */
3 Flags        fixed bin(31); /* Flags */
```

## Déclaration High Level Assembler pour MQRFH

```
MQRFH          DSECT
MQRFH_STRUCID  DS   CL4  Structure identifier
MQRFH_VERSION  DS   F    Structure version number
MQRFH_STRUCLNGTH DS   F    Total length of MQRFH including
*              NAMEVALUESTRING
MQRFH_ENCODING DS   F    Numeric encoding of data that follows
*              NAMEVALUESTRING
MQRFH_CODEDCHARSETID DS   F    Character set identifier of data that
*              follows NAMEVALUESTRING
MQRFH_FORMAT   DS   CL8  Format name of data that follows
*              NAMEVALUESTRING
MQRFH_FLAGS    DS   F    Flags
*
MQRFH_LENGTH   EQU   *-MQRFH
               ORG   MQRFH
MQRFH_AREA     DS   CL(MQRFH_LENGTH)
```

## Déclaration Visual Basic pour MQRFH

```
Type MQRFH
  StrucId      As String*4 'Structure identifier'
  Version      As Long      'Structure version number'
  StrucLength  As Long      'Total length of MQRFH including'
                'NameValueString'
  Encoding     As Long      'Numeric encoding of data that follows'
                'NameValueString'
  CodedCharSetId As Long    'Character set identifier of data that'
                'follows NameValueString'
  Format        As String*8 'Format name of data that follows'
                'NameValueString'
  Flags        As Long      'Flags'
End Type
```

### **StrucId (MQCHAR4) pour MQRFH**

Il s'agit de l'identificateur de structure des règles et de la structure d'en-tête de formatage. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQRFH\_STRUC\_ID.

La valeur doit être:

#### **ID MQRFH\_STRUC\_ID**

Identificateur des règles et de la structure d'en-tête de formatage.

Pour le langage de programmation C, la constante MQRFH\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQRFH\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQRFH**

Il s'agit du numéro de version de la structure ; la valeur doit être:

## **MQRFH\_VERSION\_1**

Règles Version-1 et structure d'en-tête de formatage.

La valeur initiale de cette zone est MQRFH\_VERSION\_1.

## **StrucLength (MQLONG) pour MQRFH**

Il s'agit de la longueur en octets de la structure MQRFH, y compris la zone *NameValueString* à la fin de la structure. La longueur n'inclut pas les données utilisateur qui suivent la zone *NameValueString*.

Pour éviter les problèmes de conversion des données utilisateur dans certains environnements, *StrucLength* doit être un multiple de quatre.

La constante suivante indique la longueur de la partie *fixe* de la structure, c'est-à-dire la longueur à l'exclusion de la zone *NameValueString* :

## **MQRFH\_STRUC\_LENGTH\_FIXED**

Longueur de la partie fixe de la structure MQRFH.

La valeur initiale de cette zone est MQRFH\_STRUC\_LENGTH\_FIXED.

## **Codage (MQLONG) pour MQRFH**

Indique le codage numérique des données qui suivent *NameValueString* ; elle ne s'applique pas aux données numériques de la structure MQRFH elle-même.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données.

La valeur initiale de cette zone est MQENC\_NATIVE.

## **CodedCharSetId (MQLONG) pour MQRFH**

Indique l'identificateur de jeu de caractères des données qui suivent *NameValueString* ; elle ne s'applique pas aux données de type caractères dans la structure MQRFH elle-même.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données. La valeur spéciale suivante peut être utilisée:

## **MQCCSI\_HÉRITER**

Les données de type caractères dans les données *suivant* cette structure se trouvent dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, la valeur MQCCSI\_INHERIT n'est pas renvoyée par l'appel MQGET.

MQCCSI\_INHERIT ne peut pas être utilisé si la valeur de la zone *PutApplType* dans MQMD est MQAT\_BROKER.

La valeur initiale de cette zone est MQCCSI\_UNDEFINED.

## **Format (MQCHAR8) pour MQRFH**

Indique le nom de format des données qui suivent *NameValueString*.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de la zone *Format* dans MQMD.

La valeur initiale de cette zone est MQFMT\_NONE.

## **Indicateurs (MQLONG) pour MQRFH**

Les éléments suivants peuvent être spécifiés:

## **MQRFH\_AUCUN**

Aucun indicateur.

La valeur initiale de cette zone est MQRFH\_NONE.

### **NameValueChaîne (MQCHARn) pour MQRFH**

Il s'agit d'une chaîne de caractères de longueur variable contenant des paires nom-valeur au format suivant:

```
name1 value1 name2 value2 name3 value3 ...
```

Chaque nom ou valeur doit être séparé du nom ou de la valeur adjacente par un ou plusieurs caractères blancs ; ces blancs ne sont pas significatifs. Un nom ou une valeur peut contenir des blancs significatifs en ajoutant en préfixe et en suffixe le nom ou la valeur avec des guillemets ; tous les caractères entre les guillemets doubles ouverts et les guillemets fermants correspondants sont considérés comme significatifs. Dans l'exemple suivant, le nom est FAMOUS\_WORDS et la valeur est Hello World:

```
FAMOUS_WORDS "Hello World"
```

Un nom ou une valeur peut contenir des caractères autres que le caractère null (qui sert de délimiteur pour *NameValueString*). Toutefois, pour faciliter l'interopérabilité, une application peut limiter les noms aux caractères suivants:

- Premier caractère: majuscule ou minuscule (A à Z ou a à z) ou trait de soulignement.
- Caractères suivants: caractères alphabétiques majuscules ou minuscules, chiffres décimaux (0 à 9), trait de soulignement, trait d'union ou point.

Si un nom ou une valeur contient un ou plusieurs guillemets, le nom ou la valeur doit être placé entre guillemets et chaque guillemet dans la chaîne doit être doublé:

```
Famous_Words "The program displayed ""Hello World"""
```

Les noms et les valeurs sont sensibles à la casse, c'est-à-dire que les minuscules ne sont pas considérées comme des majuscules. Par exemple, FAMOUS\_WORDS et Famous\_Words sont deux noms différents.

La longueur en octets de *NameValueString* est égale à *StrucLength* moins MQRFH\_STRUC\_LENGTH\_FIXED. Pour éviter les problèmes de conversion des données utilisateur dans certains environnements, faites de cette longueur un multiple de quatre. Placez *NameValueString* avec des blancs de cette longueur ou terminez-le plus tôt en plaçant un caractère nul après le dernier caractère significatif de la chaîne. Le caractère null et les octets qui le suivent, jusqu'à la longueur spécifiée de *NameValueString*, sont ignorés.

**Remarque :** Comme la longueur de cette zone n'est pas fixe, elle est omise dans les déclarations de la structure fournies pour les langages de programmation pris en charge.

## **MQRFH2 - En-tête 2 de règles et de formatage**

L'en-tête MQRFH2 est basé sur l'en-tête MQRFH , mais il permet aux chaînes Unicode d'être transportées sans traduction, et il peut contenir des types de données numériques. La structure MQRFH2 définit le format des règles version-2 et de l'en-tête de formatage. Vous utilisez cet en-tête pour envoyer des données qui ont été codées à l'aide d'une syntaxe de type XML. Un message peut contenir au moins deux structures MQRFH2 en série, les données utilisateur pouvant éventuellement suivre la dernière structure MQRFH2 de la série.

### **Disponibilité**

Tous les systèmes IBM MQ , plus IBM MQ MQI clients connectés à ces systèmes.

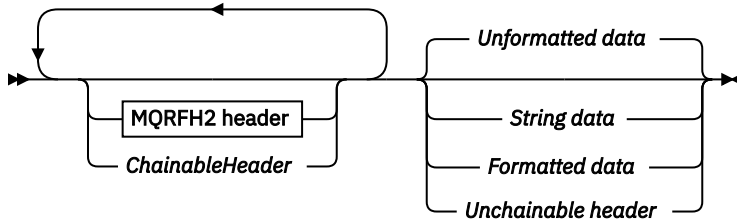
### **Format**

MQFMT\_RF\_HEADER\_2

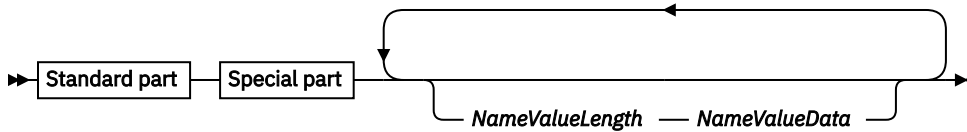


## Syntax

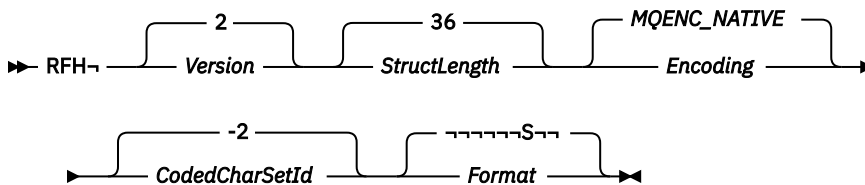
### IBM MQ Message



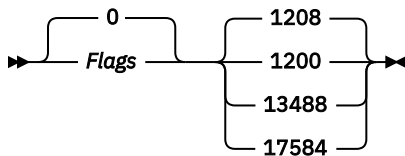
### MQRFH2 header



### Standard part



### Special part



## Jeu de caractères et codage

Des règles spéciales s'appliquent au jeu de caractères et au codage utilisés pour la structure MQRFH2 :

- Les zones autres que *NameValueData* sont dans le jeu de caractères et le codage donnés par les zones *CodedCharSetId* et *Encoding* de la structure d'en-tête qui précède MQRFH2, ou par les zones de la structure MQMD si MQRFH2 est au début des données de message d'application.

Le jeu de caractères doit comporter des caractères mono-octet pour les caractères admis dans les noms de file d'attente.

Lorsque MQGMO\_CONVERT est spécifié dans l'appel MQGET , le gestionnaire de files d'attente convertit les zones MQRFH2 , autres que *NameValueData*, en jeu de caractères et en codage requis.

- *NameValueData* correspond au jeu de caractères indiqué par la zone *NameValueCCSID* . Seuls les jeux de caractères Unicode répertoriés sont valides pour *NameValueCCSID* ; voir la description de *NameValueCCSID* pour plus de détails.

Certains jeux de caractères ont une représentation qui dépend du codage. Si *NameValueCCSID* est l'un de ces jeux de caractères, *NameValueData* doit être dans le même codage que les autres zones du MQRFH2.

Lorsque MQGMO\_CONVERT est spécifié dans l'appel MQGET , le gestionnaire de files d'attente convertit *NameValueData* au codage demandé, mais ne modifie pas son jeu de caractères.

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Tableau 514. Zones de MQRFH2 pour MQRFH2		
Nom de zone	Nom de la constante	Valeur de la constante
<u>StrucId</u> (identificateur de structure)	ID MQRFH_STRUC_ID	'RFH↵'
<u>Version</u> (numéro de version de la structure)	MQRFH_VERSION_2	2
<u>StrucLength</u> (longueur en octets de la structure MQRFH2)	MQRFH_STRUC_LENGTH_FIXED_2	36
<u>Codage</u> (codage numérique des données qui suivent la dernière zone <i>NameValueData</i> )	MQENC_NATIVE	Dépend de l'environnement
<u>CodedCharSetId</u> (identificateur de jeu de caractères des données qui suivent la dernière zone <i>NameValueData</i> )	MQCCSI_HÉRITER	-2
<u>Format</u> (nom de format des données qui suivent la dernière zone <i>NameValueData</i> )	MQFMT_AUCUN	Espaces vides
<u>Indicateurs</u> (indicateurs)	MQRFH_AUCUN	0
<u>NameValueCCSID</u> (ID de jeu de caractères codés des données de la zone <i>NameValueData</i> )	Aucun	1208
<u>NameValueLongueur</u> (longueur en octets des données de la zone <i>NameValueData</i> )	Aucun	None
<u>NameValueData</u> (paires nom-valeur des propriétés de message)	Aucun	Aucun
<p><b>Remarques :</b></p> <ol style="list-style-type: none"> <li>1. Le symbole ↵ représente un caractère blanc unique.</li> <li>2. Dans le langage de programmation C, la variable macro MQRFH2_DEFAULT contient les valeurs répertoriées dans le tableau. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</li> </ol> <pre style="background-color: #f0f0f0; padding: 5px;">MQRFH2 MyRFH2 = {MQRFH2_DEFAULT};</pre>		

## Déclarations de langue

### Déclaration C pour MQRFH2

```
typedef struct tagMQRFH2 MQRFH2;
struct tagMQRFH2 {
    MQCHAR4  StrucId;          /* Structure identifier */
    MQLONG   Version;         /* Structure version number */
    MQLONG   StrucLength;     /* Total length of MQRFH2 including all
                             NameValueLength and NameValueData
                             fields */
    MQLONG   Encoding;       /* Numeric encoding of data that follows
                             last NameValueData field */
    MQLONG   CodedCharSetId; /* Character set identifier of data that
                             follows last NameValueData field */
    MQCHAR8  Format;         /* Format name of data that follows last
                             NameValueData field */
    MQLONG   Flags;         /* Flags */
    MQLONG   NameValueCCSID; /* Character set identifier of
                             NameValueData */
};
```

### Déclaration COBOL pour MQRFH2

```
** MQRFH2 structure
10 MQRFH2.
** Structure identifier
15 MQRFH2-STRUCID PIC X(4).
** Structure version number
15 MQRFH2-VERSION PIC S9(9) BINARY.
** Total length of MQRFH2 including all NAMEVALUELENGTH and
** NAMEVALUEDATA fields
15 MQRFH2-STRUCLNGTH PIC S9(9) BINARY.
** Numeric encoding of data that follows last NAMEVALUEDATA field
15 MQRFH2-ENCODING PIC S9(9) BINARY.
** Character set identifier of data that follows last NAMEVALUEDATA
** field
15 MQRFH2-CODEDCHARSETID PIC S9(9) BINARY.
** Format name of data that follows last NAMEVALUEDATA field
15 MQRFH2-FORMAT PIC X(8).
** Flags
15 MQRFH2-FLAGS PIC S9(9) BINARY.
** Character set identifier of NAMEVALUEDATA
15 MQRFH2-NAMEVALUECCSID PIC S9(9) BINARY.
```

### Déclaration PL/I pour MQRFH2

```
dcl
1 MQRFH2 based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 StrucLength fixed bin(31), /* Total length of MQRFH2 including
                             all NameValueLength and
                             NameValueData fields */
3 Encoding fixed bin(31), /* Numeric encoding of data that
                             follows last NameValueData field */
3 CodedCharSetId fixed bin(31), /* Character set identifier of data
                             that follows last NameValueData
                             field */
3 Format char(8), /* Format name of data that follows
                             last NameValueData field */
3 Flags fixed bin(31), /* Flags */
3 NameValueCCSID fixed bin(31); /* Character set identifier of
                             NameValueData */
```

### Déclaration High Level Assembler pour MQRFH2

MQRFH	DSECT		
MQRFH_STRUCID	DS	CL4	Structure identifier
MQRFH_VERSION	DS	F	Structure version number
MQRFH_STRUCLNGTH	DS	F	Total length of MQRFH2 including all NAMEVALUELENGTH and NAMEVALUEDATA fields
* MQRFH_ENCODING	DS	F	Numeric encoding of data that follows

*	MQRFH_CODEDCHARSETID	DS	F	last NAMEVALUEDATA field
*				Character set identifier of data that follows last NAMEVALUEDATA field
*	MQRFH_FORMAT	DS	CL8	Format name of data that follows last NAMEVALUEDATA field
*				NAMEVALUEDATA field
*	MQRFH_FLAGS	DS	F	Flags
*	MQRFH_NAMEVALUECCSID	DS	F	Character set identifier of NAMEVALUEDATA
*				NAMEVALUEDATA
*	MQRFH_LENGTH	EQU	*-MQRFH	
		ORG	MQRFH	
	MQRFH_AREA	DS	CL(MQRFH_LENGTH)	

## Déclaration Visual Basic pour MQRFH2

```

Type MQRFH2
  StrucId      As String*4 'Structure identifier'
  Version     As Long      'Structure version number'
  StrucLength  As Long      'Total length of MQRFH2 including all'
                                     'NameValueLength and NameValueData fields'
  Encoding     As Long      'Numeric encoding of data that follows'
                                     'last NameValueData field'
  CodedCharSetId As Long    'Character set identifier of data that'
                                     'follows last NameValueData field'
  Format       As String*8  'Format name of data that follows last'
                                     'NameValueData field'
  Flags       As Long      'Flags'
  NameValueCCSID As Long    'Character set identifier of NameValueData'
End Type

```

### **StrucId (MCHAR4) pour MQRFH2**

Il s'agit de l'identificateur de structure des règles et de l'en-tête de formatage de deux structures. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQRFH2\_STRUC\_ID.

La valeur doit être:

#### **MQRFH2\_STRUC\_ID**

Identificateur des règles et de la structure d'en-tête de formatage deux.

Pour le langage de programmation C, la constante MQRFH2\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQRFH2\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQRFH2**

Il s'agit du numéro de version de la structure ; la valeur doit être:

#### **MQRFH\_VERSION\_2**

Règles Version-2 et structure d'en-tête de formatage.

La valeur initiale de cette zone est MQRFH\_VERSION\_2.

### **StrucLength (MQLONG) pour MQRFH2**

Il s'agit de la longueur en octets de la structure MQRFH2 , y compris les zones *NameValueLength* et *NameValueData* à la fin de la structure. Il est possible qu'il y ait plusieurs paires de zones *NameValueLength* et *NameValueData* à la fin de la structure, dans la séquence:

```
length1, data1, length2, data2, ...
```

*StrucLength* n'inclut pas les données utilisateur qui peuvent suivre la dernière zone *NameValueData* à la fin de la structure.

Pour éviter les problèmes liés à la conversion des données utilisateur dans certains environnements, *StrucLength* doit être un multiple de quatre.

La constante suivante indique la longueur de la partie *fixe* de la structure, c'est-à-dire la longueur à l'exclusion des zones *NameValueLength* et *NameValueData* :

## **MQRFH\_STRUC\_LENGTH\_FIXED\_2**

Longueur de la partie fixe de la structure MQRFH2 .

La valeur initiale de cette zone est MQRFH\_STRUC\_LENGTH\_FIXED\_2.

## **Codage (MQLONG) pour MQRFH2**

Indique le codage numérique des données qui suivent la dernière zone *NameValueData* ; il ne s'applique pas aux données numériques de la structure MQRFH2 elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données.

La valeur initiale de cette zone est MQENC\_NATIVE.

## **CodedCharSetId (MQLONG) pour MQRFH2**

Indique l'identificateur de jeu de caractères des données qui suivent la dernière zone *NameValueData* ; il ne s'applique pas aux données de type caractère de la structure MQRFH2 elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. La valeur spéciale suivante peut être utilisée:

### **MQCCSI\_HÉRITER**

Les données de type caractères dans les données *suivant* cette structure se trouvent dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, la valeur MQCCSI\_INHERIT n'est pas renvoyée par l'appel MQGET.

MQCCSI\_INHERIT ne peut pas être utilisé si la valeur de la zone *PutApplType* dans MQMD est MQAT\_BROKER.

La valeur initiale de cette zone est MQCCSI\_INHERIT.

## **Format (MQCHAR8) pour MQRFH2**

Indique le nom de format des données qui suivent la dernière zone *NameValueData* .

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de la zone *Format* dans MQMD.

La valeur initiale de cette zone est MQFMT\_NONE.

## **Indicateurs (MQLONG) pour MQRFH2**

La valeur initiale de cette zone est MQRFH\_NONE. MQRFH\_NONE doit être précisé.

### **MQRFH\_NONE**

Aucun indicateur.

### **MQRFH\_INTERNAL**

L'en-tête MQRFH2 contient des propriétés définies en interne.

MQRFH\_INTERNAL est destiné à être utilisé par le gestionnaire de files d'attente.

Les 16 premiers bits, MQRFH\_FLAGS\_RESTRICTED\_MASK, sont réservés aux indicateurs des ensembles de gestionnaires de files d'attente. Les indicateurs qu'un utilisateur peut définir sont définis dans les 16 derniers bits.

## **NameValueCCSID (MQLONG) pour MQRFH2**

Indique l'identificateur de jeu de caractères codés des données de la zone *NameValueData* . Il est différent du jeu de caractères des autres chaînes de la structure MQRFH2 et peut être différent du jeu de caractères des données (le cas échéant) qui suit la dernière zone *NameValueData* à la fin de la structure.

*NameValueCCSID* doit avoir l'une des valeurs suivantes:

CCSID	Explication
1200	UTF-16, version Unicode la plus récente prise en charge
13488	UTF-16, sous-ensemble de la version Unicode 2.0
17584	UTF-16, Unicode version 3.0 sous-ensemble (inclut le symbole Euro)
1208	UTF-8, version Unicode la plus récente prise en charge

Pour les jeux de caractères UTF-16 , le codage (ordre des octets) de *NameValueData* doit être identique au codage des autres zones de la structure *MQRFH2* .

Les caractères au-delà du plan multilingue de base Unicode (ceux au-dessus de U + FFFF), représentés en UTF-16 par des points de code de substitution (X'D800'à X'DFFF'), ou quatre octets en UTF-8, ne sont pas pris en charge.

**Remarque :** Si *NameValueCCSID* ne possède pas l'une des valeurs répertoriées ci-dessus et que la structure *MQRFH2* requiert une conversion sur l'appel *MQGET*, l'appel se termine avec le code anomalie *MQRC\_SOURCE\_CCSID\_ERROR* et le message est renvoyé non converti.

La valeur initiale de cette zone est 1208.

### ***NameValueLongueur (MQLONG) pour MQRFH2***

Longueur de la zone *NameValueData* correspondante

Indique la longueur en octets des données de la zone *NameValueData* . *NameValueLength* doit être un multiple de quatre.

**Remarque :** Les zones *NameValueLength* et *NameValueData* sont facultatives, mais si elles sont présentes, elles doivent apparaître sous la forme d'une paire et être adjacentes. La paire de champs peut être répétée autant de fois que nécessaire, par exemple:

```
length1 data1 length2 data2 length3 data3
```

Etant donné que ces zones sont facultatives, elles sont omises dans les déclarations de la structure fournies pour les différents langages de programmation pris en charge.

### ***NameValueData (MQCHARn) pour MQRFH2***

*NameValueData* est une zone de longueur variable qui contient un dossier contenant des paires nom-valeur de propriétés de message. Un dossier est une chaîne de caractères de longueur variable contenant des données codées à l'aide d'une syntaxe de type XML. La longueur en octets de la chaîne de caractères est donnée par la zone *NameValueLength* qui précède la zone *NameValueData* . La longueur doit être un multiple de quatre.

Les zones *NameValueLength* et *NameValueData* sont facultatives, mais si elles sont présentes, elles doivent apparaître sous la forme d'une paire et être adjacentes. La paire de champs peut être répétée autant de fois que nécessaire, par exemple:

```
length1 data1 length2 data2 length3 data3
```

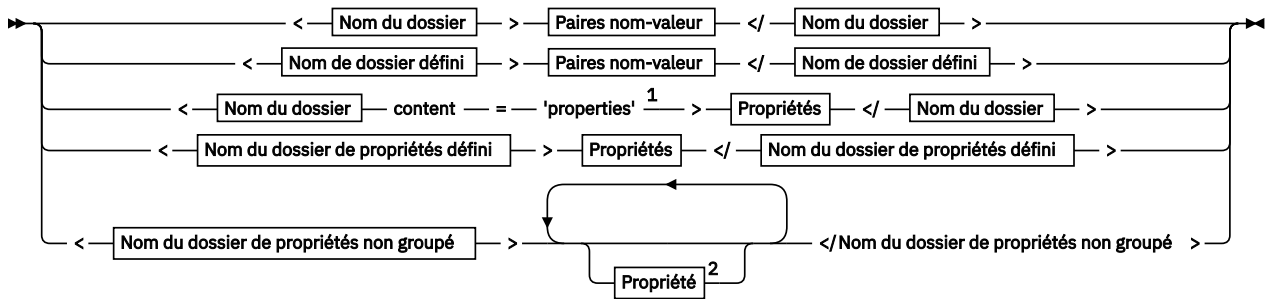
*NameValueData* n'est pas converti au jeu de caractères spécifié dans l'appel *MQGET* . Même si le message est extrait avec l'option *MQGMO\_CONVERT* en vigueur, *NameValueData* reste dans son jeu de caractères d'origine. Toutefois, *NameValueData* est converti au codage spécifié dans l'appel *MQGET* .

#### **Remarques :**

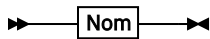
- Etant donné que ces zones sont facultatives, elles sont omises dans les déclarations de la structure fournies pour les différents langages de programmation pris en charge.

- Les termes "défini" et "réservé" sont utilisés dans le diagramme de syntaxe. "Défini" signifie que le nom est utilisé par IBM MQ. "Réservé" signifie que le nom est réservé pour une utilisation ultérieure par IBM MQ.

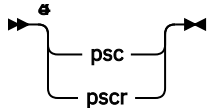
### NameValueData Syntaxe



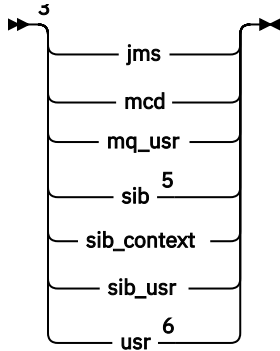
#### Nom du dossier



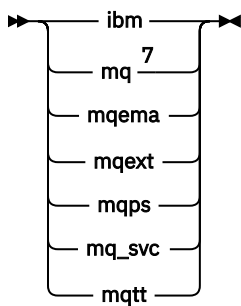
#### Nom de dossier défini



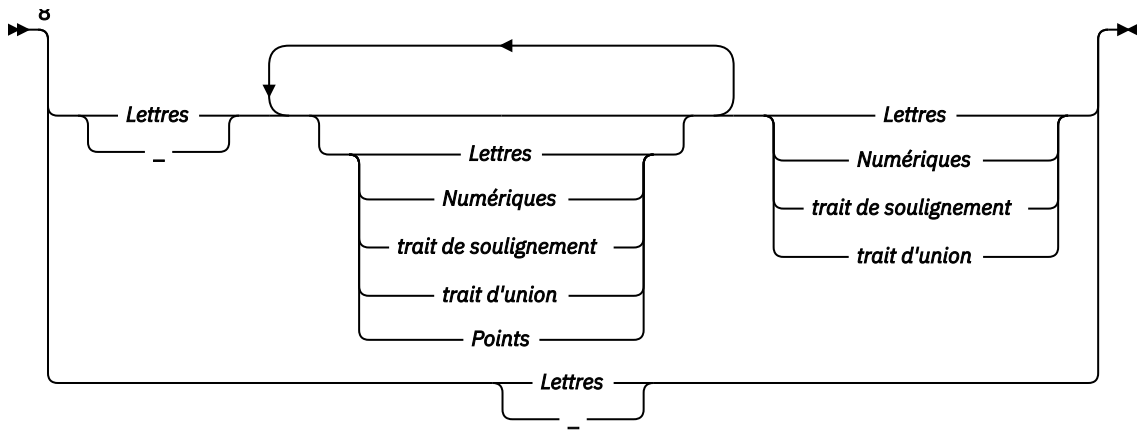
#### Nom du dossier de propriétés défini



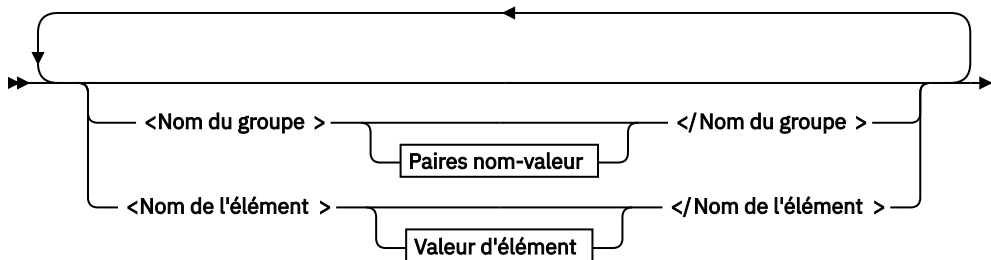
#### Nom du dossier de propriétés non groupé



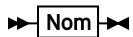
#### Nom



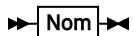
**Paires nom-valeur**



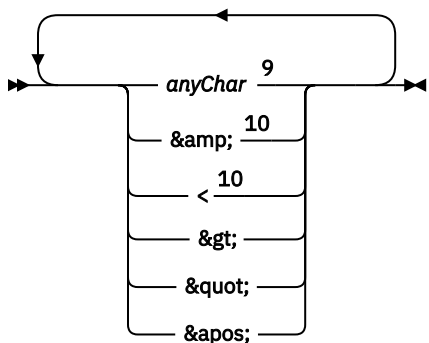
**Nom du groupe**



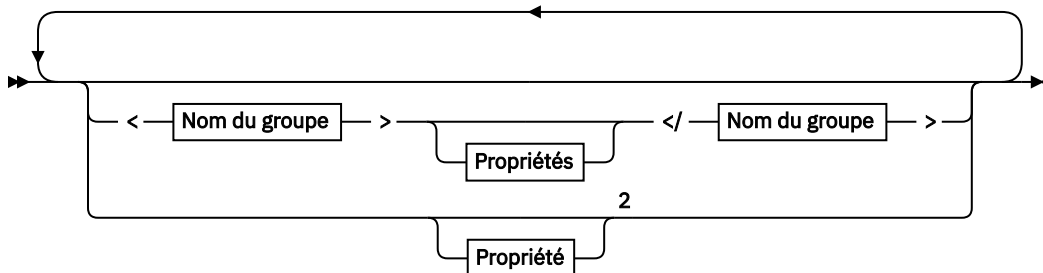
**Nom de l'élément**



**Valeur d'élément**

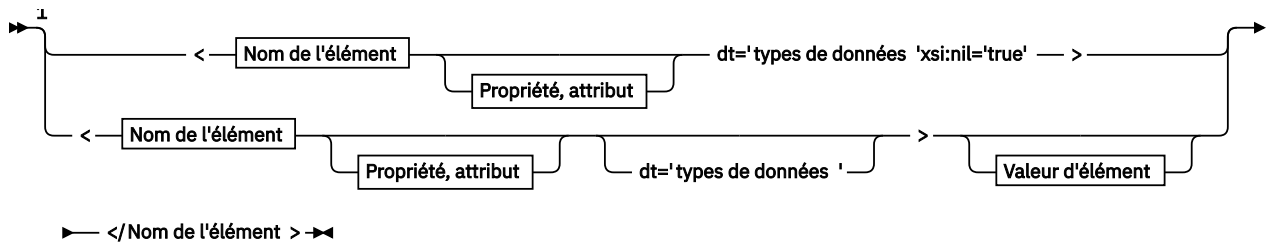


**Propriétés**

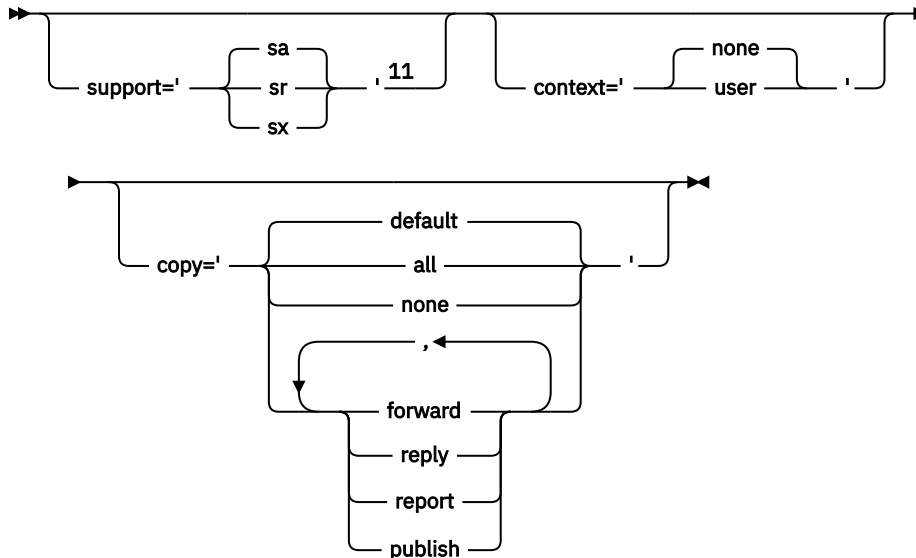


**Propriété**

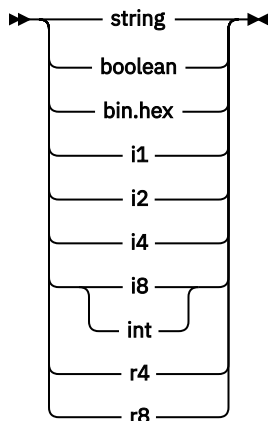




### Propriété, attribut



### types de données



### Remarques :

- <sup>1</sup> Les guillemets ou les apostrophes sont valides.
- <sup>2</sup> N'utilisez pas de nom de propriété non valide ; voir «Nom de propriété non valide», à la page 573. Utilisez un nom de propriété réservée uniquement à des fins définies ; voir «Noms de propriété définis», à la page 573.
- <sup>3</sup> Le nom doit être en minuscules.
- <sup>4</sup> Un seul dossier `psc` et `psc:` est pris en charge.
- <sup>5</sup> WebSphere Application Server Service Integration Bus ignore les dossiers `sib`, `sib_context` et `sib_usr` dans les en-têtes MQRFH2 suivants, et seules les propriétés du premier en-tête MQRFH2 sont significatives.
- <sup>6</sup> Un seul dossier `usr` ne doit pas être présent dans un MQRFH2. Les propriétés du dossier `usr` ne doivent pas apparaître plus d'une fois.

<sup>7</sup> Seules les propriétés du premier dossier mq sont significatives. Si le dossier est UTF -8, seuls les caractères UTF -8 mono-octet sont pris en charge. Le seul caractère blanc est Unicode U+0020.

<sup>8</sup> Les caractères valides sont définis dans la spécification XML W3C et consistent essentiellement en des catégories Unicode Ll, Lu, Lo, Lt, Nl, Mc, Mn, Lm, et Nd ; voir [Catégories de caractères Unicode](#).

<sup>9</sup> Tous les caractères sont significatifs. Les blancs de début et de fin font partie de la valeur de l'élément.

<sup>10</sup> N'utilisez pas de caractère non valide ; voir «Caractères non valides», à la page 573. Utilisez une séquence d'échappement plutôt que ces caractères non valides.

<sup>11</sup> L'attribut de propriété de support est valide uniquement sur le dossier mq

## Nom du dossier

*NameValueData* contient un seul dossier. Pour créer plusieurs dossiers, créez plusieurs zones *NameValueData* . Vous pouvez créer plusieurs zones *NameValueData* dans un même en-tête MQRFH2 dans un message. Vous pouvez également créer plusieurs en-têtes MQRFH2 chaînés, chacun contenant plusieurs zones *NameValueData* .

L'ordre des en-têtes MQRFH2 et l'ordre des zones *NameValueData* ne font aucune différence avec le contenu logique d'un dossier. Si le même dossier est présent plusieurs fois dans un message, le dossier est analysé comme un tout. Si la même propriété se produit dans plusieurs instances du même dossier, elle est analysée en tant que liste.

Une analyse correcte d'un MQRFH2 n'est pas affectée par les autres manières dont un dossier peut être physiquement stocké dans un message.

Quatre dossiers ne respectent pas cette règle. Seule la première instance du dossier mq, sib, sib\_contextet sib\_usr est analysée.

Si la même propriété apparaît plusieurs fois dans le contenu combiné des en-têtes MQRFH2 chaînés, seule la première instance de la propriété est analysée. Si une propriété est définie à l'aide d'un appel d'API, tel que MQSETMP, et ajoutée à un MQRFH2 directement par une application, l'appel d'API est prioritaire.

Un nom de dossier est le nom d'un dossier contenant des paires nom-valeur ou des groupes. Les groupes et les paires nom-valeur peuvent être mélangés au même niveau dans l'arborescence des dossiers ; voir [Figure 1](#), à la page 562. Ne combinez pas un nom de groupe et un nom d'élément ; voir [Figure 2](#), à la page 562

---

```
<group1><nvp1>value</nvp1></group1><group2><nvp2>value</nvp2></group2>  
<group3><nvp1>value</nvp1></group3><nvp3>value</nvp3>
```

*Figure 1. Utilisation correcte des groupes et des paires nom-valeur*

---

```
<group1><nvp1> value </nvp1> value </group1>
```

*Figure 2. Utilisation incorrecte des groupes et des paires nom-valeur*

---

N'utilisez pas de nom de dossier non valide ou réservé ; voir «Nom de chemin d'accès non valide», à la page 572 et «Nom du dossier réservé ou du dossier de propriétés», à la page 572. Utilisez un nom de dossier défini uniquement à des fins définies ; voir «Nom de dossier défini», à la page 563.

Si vous ajoutez l'attribut ' content=properties ' à la balise de nom de dossier, le dossier devient un dossier de propriétés ; voir [Figure 3](#), à la page 563.

---

```
<myFolder></myFolder>
<myPropertyFolder contents='properties'></myPropertyFolder>
```

Figure 3. Exemple de dossier et de dossier de propriétés

---

Les noms de dossier sont sensibles à la casse. Les noms de dossier et les noms de dossier de propriété partagent le même espace de nom. Ils doivent avoir des noms différents. Folder1 dans [Figure 4](#), à la page 563 doit être un nom différent de Folder2 dans [Figure 5](#), à la page 563.

---

```
< Folder1 ><NVP1> value </NVP1></ Folder1 >
```

Figure 4. Espace de nom Folder1

---

```
< Folder2 content='properties'>< Property1 > value </ Property1 ></ Folder2 >
```

Figure 5. Espace de nom Folder2

---

Les groupes, les propriétés et les paires nom-valeur des différents dossiers ont des espaces de nom différents. Property1 dans [Figure 5](#), à la page 563 est une propriété différente de Property1 dans [Figure 6](#), à la page 563.

---

```
<Folder3 content='properties'>< Property1 > value </ Property1 ></Folder3>
```

Figure 6. Espace de nom Folder3

---

Les dossiers de propriétés sont différents des dossiers autres que des dossiers de propriétés à deux égards importants:

1. Les dossiers de propriétés contiennent des propriétés et les dossiers autres que les dossiers de propriétés contiennent des paires nom-valeur. Les dossiers diffèrent légèrement, du point de vue de la syntaxe.
2. Utilisez les interfaces définies, telles que les propriétés MQI ou les propriétés de message JMS , pour accéder aux propriétés de message. Les interfaces garantissent que les dossiers de propriétés dans MQRFH2 sont syntaxiquement appropriés. Un dossier de propriétés syntaxiquement correct est interopérable entre les gestionnaires de files d'attente sur différentes plateformes et différentes versions.

La propriété de message MQI est une méthode robuste de lecture et d'écriture d'un MQRFH2 et évite les difficultés d'analyse syntaxique correcte d'un MQRFH2 .

## Nom de dossier défini

Un nom de dossier défini est le nom d'un dossier réservé à IBM MQ ou à un autre produit. Ne créez pas de dossier du même nom et n'ajoutez pas vos propres paires nom-valeur aux dossiers. Les dossiers définis sont psc et pscr.

psc et pscr sont utilisés par la publication / l'abonnement en file d'attente.

Un message segmenté inséré avec MQMF\_SEGMENT ou MQMF\_SEGMENTATION\_ALLOWED ne peut pas contenir de MQRFH2 avec un nom de dossier défini. Le MQPUT échoue avec le code anomalie 2443, MQRC\_SEGMENTATION\_NOT\_ALLOWED.

## Nom du dossier de propriétés défini

Un nom de dossier de propriétés défini est le nom d'un dossier de propriétés utilisé par IBM MQ ou un autre produit. Pour les noms des dossiers et leur contenu, voir [Dossiers de propriétés](#). Les noms de dossier de propriétés définis sont un sous-ensemble de tous les noms de dossier réservés par IBM MQ; voir «Nom du dossier réservé ou du dossier de propriétés», à la page 572.

Tout élément stocké dans un dossier de propriétés défini est une propriété. Un élément stocké dans un dossier de propriétés défini ne doit pas avoir d'attribut content= 'properties' .

Vous pouvez ajouter des propriétés uniquement aux dossiers de propriétés définis `usr`, `mq_usret` `sib_usr`. Dans d'autres dossiers de propriétés, tels que `mq` et `sib`, IBM MQ ignore ou rejette les propriétés qu'il ne reconnaît pas.

La description de chaque dossier de propriétés défini répertorie les propriétés définies par IBM MQ qui peuvent être utilisées par les programmes d'application. Certaines des propriétés sont accessibles indirectement en définissant ou en obtenant une propriété JMS , et d'autres sont accessibles directement à l'aide des appels MQSETMP et MQINQMP MQI.

Les dossiers de propriétés définies contiennent également d'autres propriétés réservées par IBM MQ , mais auxquelles les applications n'ont pas accès. Les noms des propriétés réservées ne sont pas répertoriés. Aucune propriété réservée n'est présente dans les dossiers de propriétés `usr`, `mq_usret` `sib_usr` . Mais ne créez pas de propriétés avec des noms de propriété non valides ; voir «Nom de propriété non valide», à la page 573.

## Dossiers de propriétés

### jms

`jms` contient des zones d'en-tête JMS et des propriétés JMSX qui ne peuvent pas être entièrement exprimées dans MQMD. Le dossier `jms` est toujours présent dans un MQRFH2JMS.

Tableau 515. Nom de propriété jms, synonyme, type de données et dossier			
Synonyme de propriété	Nom de la propriété	Type de données	Dossier
JMSDestination	<code>jms.Dst</code>	string	<code>&lt;jms&gt;&lt;Dst&gt; destination &lt;/Dst&gt;&lt;/jms&gt;</code>
JMSExpiration	<code>jms.Exp</code>	i8	<code>&lt;jms&gt;&lt;Exp&gt; expiration &lt;/Exp&gt;&lt;/jms&gt;</code>
corrélation JMS	<code>jms.Cid</code>	string	<code>&lt;jms&gt;&lt;Cid&gt; correlationId &lt;/Cid&gt;&lt;/jms&gt;</code>
JMSDelivery	<code>jms.Dlv</code>	i4	<code>&lt;jms&gt;&lt;Dlv&gt; delivery &lt;/Dlv&gt;&lt;/jms&gt;</code>
JMSPriority	<code>jms.Pri</code>	i4	<code>&lt;jms&gt;&lt;Pri&gt; priority &lt;/Pri&gt;&lt;/jms&gt;</code>
JMSReplyTo	<code>jms.Rto</code>	string	<code>&lt;jms&gt;&lt;Rto&gt; replyToURI &lt;/Rto&gt;&lt;/jms&gt;</code>
JMSTimestamp	<code>jms.Tms</code>	i8	<code>&lt;jms&gt;&lt;Tms&gt; timestamp &lt;/Tms&gt;&lt;/jms&gt;</code>

Tableau 515. Nom de propriété jms, synonyme, type de données et dossier (suite)			
Synonyme de propriété	Nom de la propriété	Type de données	Dossier
JMSXGroupID	.jms.Gid	string	<jms><Gid> <i>groupId</i> </Gid></jms>
JMSXGroupSeq	.jms.Seq	i4	<jms><Seq> <i>messageSequenceNo</i> </Seq></jms>

N'ajoutez pas vos propres propriétés dans le dossier jms.

### mcd

mcd contient des propriétés qui décrivent le format du message. Par exemple, la propriété du domaine de service de message Msd identifie un message JMS comme étant JMSTextMessage, JMSBytesMessage, JMSStreamMessage, JMSMapMessage, JMSObjectMessage ou NULL.

Le dossier mcd est toujours présent dans un message JMS contenant un MQRFH2.

Il est toujours présent dans un message contenant un MQRFH2 envoyé par IBM Integration Bus. Il décrit le domaine, le format, le type et l'ensemble de messages d'un message.

Tableau 516. Nom de la propriété mcd, synonyme, type de données et dossier			
Synonyme de propriété	Nom de la propriété	Type de données	Dossier
	mcd.Msd	string	<mcd><Msd> <i>messageDomain</i> </Msd></mcd>
	mcd.Set	string	<mcd><Set> <i>messageDomain</i> </Set></mcd>
	mcd.Type	string	<mcd><Type> <i>messageDomain</i> </Type></mcd>
	mcd.Fmt	string	<mcd><Fmt> <i>messageDomain</i> </Fmt></mcd>

N'ajoutez pas vos propres propriétés dans le dossier mcd.

### mq\_usr

mq\_usr contient des propriétés définies par l'application qui ne sont pas exposées en tant que propriétés définies par l'utilisateur JMS. Les propriétés qui ne répondent pas aux exigences de JMS peuvent être placées dans ce dossier.

Vous pouvez créer des propriétés dans le dossier mq\_usr. Les propriétés que vous créez dans le mq\_usr sont des propriétés que vous créez dans de nouveaux dossiers avec l'attribut content='properties'.

### sib

sib contient les propriétés des messages système du bus d'intégration de services WebSphere Application Server (WAS/SIB). Les propriétés sib ne sont pas exposées en tant que propriétés JMS aux applications IBM MQ JMS car elles ne sont pas des types pris en charge. Par exemple, certaines propriétés sib ne peuvent pas être exposées en tant que propriétés JMS car il s'agit de tableaux d'octets. Certaines propriétés sib sont exposées aux applications WAS/SIB en tant que

propriétés JMS\_IBM\_\* ; elles incluent les propriétés des chemins de routage de réacheminement et de réacheminement inverse.

N'ajoutez pas vos propres propriétés dans le dossier sib.

### sib\_context

sib\_context contient des propriétés de message système WAS/SIB qui ne sont pas exposées aux applications utilisateur WAS/SIB ou en tant que propriétés JMS . sib\_context contient les propriétés de sécurité et transactionnelles utilisées pour les services Web.

N'ajoutez pas vos propres propriétés dans le dossier sib\_context.

### sib\_usr

sib\_usr contient des propriétés de message utilisateur WAS/SIB qui ne sont pas exposées en tant que propriétés utilisateur JMS car elles ne sont pas de types pris en charge. sib\_usr est exposé aux applications WAS/SIB dans l'interface SIMessage ; voir [Développement d'une intégration de services](#).

Le type d'une propriété sib\_usr doit être bin.hex et la valeur doit être au format correct. Si une application IBM MQ écrit un élément de type bin.hex dans le dossier dans un format incorrect, l'application reçoit un IOException. Si le type de données de la propriété n'est pas bin.hex, l'application reçoit un ClassCastException.

N'essayez pas de rendre les propriétés utilisateur JMS disponibles pour WAS/SIB à l'aide de ce dossier ; utilisez plutôt le dossier usr.

Vous pouvez créer des propriétés dans le dossier sib\_usr.

### usr

usr contient les propriétés JMS définies par l'application associées au message. Le dossier usr n'est présent que si une application a défini une propriété définie par l'application.

usr est le dossier de propriétés par défaut. Si une propriété est définie sans nom de dossier, elle est placée dans le dossier usr.

<i>Tableau 517. Nom de la propriété usr, synonyme, type de données et dossier</i>			
<b>Synonyme de propriété</b>	<b>Nom de la propriété</b>	<b>Type de données</b>	<b>Dossier</b>
	usr.contentType	string	<usr><contentType>text/xml; charset=utf-8</contentType></usr>
	usr.endpointURL	string	<usr><endpointURL> URI </endpointURL></usr>
	usr.targetService	string	<usr><targetService> serviceName </targetService></usr>
	usr.soapAction	string	<usr><soapAction> name </soapAction></usr>
	usr.transportVersion	string	<usr><transportVersion> version </transportVersion></usr>

Vous pouvez créer des propriétés dans le dossier usr.

Un message segmenté inséré avec MQMF\_SEGMENT ou MQMF\_SEGMENTATION\_ALLOWED ne peut pas contenir de MQRFH2 avec un nom de dossier de propriété défini. Le MQPUT échoue avec le code anomalie 2443, MQRC\_SEGMENTATION\_NOT\_ALLOWED.

## Nom du dossier de propriétés non groupé

### ibm

ibm contient des propriétés qui sont utilisées uniquement par IBM MQ.

Synonyme de propriété	Nom de la propriété	Type de données	Dossier
	ibm.rfp	string	<ibm><rfp>fingerprint</rfp></ibm>

N'ajoutez pas vos propres propriétés dans le dossier ibm.

### mq

mq contient des propriétés qui sont utilisées uniquement par IBM MQ.

Les restrictions suivantes s'appliquent aux propriétés du dossier mq :

- Seules les propriétés du premier dossier mq significatif du message sont traitées par MQ; les propriétés de tout autre dossier mq du message sont ignorées.
- Seuls les caractères UTF-8 mono-octet sont autorisés dans le dossier. Un caractère multi-octet dans le dossier peut entraîner l'échec de l'analyse syntaxique et le rejet du message.
- N'utilisez pas de chaînes d'échappement dans le dossier. Une chaîne d'échappement est traitée comme la valeur réelle de l'élément.
- Seul le caractère Unicode U+0020 est traité comme un espace dans le dossier. Tous les autres caractères sont traités comme significatifs et peuvent entraîner l'échec de l'analyse syntaxique du dossier et le rejet du message.

Si l'analyse du dossier mq échoue ou si le dossier ne respecte pas ces restrictions, le message est rejeté avec le code anomalie 2527, MQRC\_RFH\_RESTRICTED\_FORMAT\_ERR.

N'ajoutez pas vos propres propriétés dans le dossier mq.

### mqema

mqema contient des propriétés qui sont utilisées uniquement par WebSphere Application Server. Le dossier a été remplacé par mqext.

N'ajoutez pas vos propres propriétés dans le dossier mqema.

### mqext

mqext contient les types de propriété suivants :

- Les propriétés qui ne sont utilisées que par WebSphere Application Server.
- Les propriétés liées à la distribution retardée des messages.

Le dossier est présent si l'application a défini au moins l'une des propriétés définies par IBM ou utilisé un retard de distribution.

Synonyme de propriété	Nom de la propriété	Type de données	Dossier
JMSArmCorrelator	mqext.Arm	string	<mqext><Arm>armCorrelator</Arm></mqext>

<i>Tableau 519. Nom de la propriété mqext, synonyme, type de données et dossier (suite)</i>			
<b>Synonyme de propriété</b>	<b>Nom de la propriété</b>	<b>Type de données</b>	<b>Dossier</b>
JMSRMCorrelator	mqext.Wrm	string	<mqext><Wrm>wrmCorrelator</Wrm></mqext>
JMSDeliveryTime	mqext.Dlt	i8	<mqext><Dlt>DeliveryTime</Dlt></mqext>
JMSDeliveryDelay	mqext.Dly	i8	<mqext><Dly>DeliveryTime</Dly></mqext>

N'ajoutez pas vos propres propriétés dans le dossier mqext.

### **mqps**

mqps contient des propriétés qui sont utilisées uniquement par la publication/l'abonnement IBM MQ. Le dossier est présent uniquement si l'application a défini au moins l'une des propriétés de publication/abonnement intégré.

<i>Tableau 520. Nom de la propriété mqps, synonyme, type de données et dossier</i>			
<b>Synonyme de propriété</b>	<b>Nom de la propriété</b>	<b>Type de données</b>	<b>Dossier</b>
MQTopicString	mqps.Top	string	<mqps><Top>topicString</Top></mqps>
MQSubscriberData	mqps.Sud	string	<mqps><Sud>subscriberUserData...</Sud></mqps>
MQIsRetained	mqps.Ret	boolean	<mqps><Ret>isRetained</Ret></mqps>
MQPubOptions	mqps.Pub	i8	<mqps><Pub>publicationOptions</Pub></mqps>
MQPubLevel	mqps.Pbl	i8	<mqps><Pbl>publicationLevel</Pbl></mqps>
MQPubTime	mqpse.Pts	string	<mqps><Pts>publicationTime</Pts></mqps>
MQPubSeqNum	mqpse.Seq	i8	<mqps><Seq>publicationSequenceNumber</Seq></mqps>
MQPubStrInpData	mqpse.Sid	string	<mqps><Sid>publicationData</Sid></mqps>
MQPubFormat	mqpse.Pfmt	i8	<mqps><Pfmt>messageFormat</Pfmt></mqps>

N'ajoutez pas vos propres propriétés dans le dossier mqps.

### **mq\_svc**

mq\_svc contient les propriétés utilisées par SupportPac MA93.

N'ajoutez pas vos propres propriétés dans le dossier mq\_svc.

### **mqtt**

mqtt contient les propriétés utilisées par MQ Telemetry



Tableau 521. Nom de propriété mqtt, synonyme, type de données et dossier

Synonyme de propriété	Nom de la propriété	Type de données	Dossier
	mqtt.clientId	string	<mqtt><clientId> <i>topicString</i> </clientId></mqtt>
	mqtt.qos	i4	<mqtt><qos> <i>qualityOfService</i> </qos></mqtt>
	mqtt.msgid	string	<mqtt><msgid> <i>messageIdentifier</i> </msgid></mqtt>

N'ajoutez pas vos propres propriétés dans le dossier mqtt.

Un message segmenté inséré avec MQMF\_SEGMENT ou MQMF\_SEGMENTATION\_ALLOWED ne peut pas contenir de MQRFH2 avec un nom de dossier de propriétés non groupé. Le MQPUT échoue avec le code anomalie 2443, MQRC\_SEGMENTATION\_NOT\_ALLOWED.

## Paired nom-valeur

Dans le diagramme de syntaxe, "paired nom-valeur" décrit le contenu d'un dossier ordinaire. Un dossier ordinaire contient des groupes et des éléments. Un élément est une paire nom-valeur. Un groupe contient des éléments et d'autres groupes.

En termes d'arbres, les éléments sont des noeuds feuille et les groupes sont des noeuds internes. Un noeud interne et le dossier, qui est le noeud racine, peuvent contenir un mélange de noeuds internes et de noeuds feuilles. Un noeud ne peut pas être à la fois un noeud interne et un noeud feuille ; voir [Figure 2](#), à la page 562.

## Propriétés

Dans le diagramme de syntaxe, "Propriétés" décrit le contenu d'un dossier de propriétés. Un dossier de propriétés contient des groupes et des propriétés. Une propriété est une paire nom-valeur avec un attribut de type de données facultatif. Un groupe contient des propriétés et d'autres groupes.

En termes d'arbres, les propriétés sont des noeuds feuilles et les groupes sont des noeuds internes. Un noeud interne et le dossier de propriétés, qui est le noeud racine, peuvent contenir un mélange de noeuds internes et de noeuds feuilles. Un noeud ne peut pas être à la fois un noeud interne et un noeud feuille ; voir [Figure 2](#), à la page 562.

## Propriété

Une propriété de message est une paire nom-valeur dans un dossier de propriétés. Il peut éventuellement inclure un attribut de type de données et un attribut de propriété ; pour un exemple, voir le code suivant. Si l'attribut de type de données est omis, le type de propriété est string.

```
<pf><p1 dt='i8' > value </p1></pf>
```

Le nom d'une propriété de message est son nom de chemin complet, avec la syntaxe <> de type XML, remplacée par des points. Par exemple, myPropertyFolder1.myGroup1.myGroup2.myProperty1 est mappé à une chaîne *NameValueData* comme suit. La chaîne est formatée pour faciliter la lecture.

```
<myPropertyFolder1>
  <myGroup1>
    <myGroup2>
      <myProperty1>value</myProperty1>
    </myGroup2>
  </myGroup1>
</myPropertyFolder1>
```

Un dossier de propriétés peut contenir plusieurs propriétés. Par exemple, les propriétés dans [Figure 7](#), à la [page 570](#) sont mappées au dossier de propriétés dans Modifications pour isoler la file d'attente des ventes dans un nouveau cluster et séparer les files d'attente de transmission du cluster de passerelle.

---

```
myPropertyFolder1.myProperty4
myPropertyFolder1.myGroup1.myGroup2.myProperty1
myPropertyFolder1.myGroup1.myGroup2.myProperty2
myPropertyFolder1.myGroup1.myProperty3
```

Figure 7. Plusieurs propriétés avec le même nom de racine

---

```
<myPropertyFolder1>
  <myProperty4>value</myProperty4>
  <myGroup1>
    <myGroup2>
      <myProperty1>value</myProperty1>
      <myProperty2>value</myProperty2>
    </myGroup2>
    <myProperty3>value</myProperty3>
  </myGroup1>
</myPropertyFolder1>
```

Figure 8. Mappage de noms de propriété multiples

---

## Nom

Un nom doit commencer par une *lettre* ou un *soulignement*. Il ne doit pas contenir de *Colon*, ne doit pas se terminer par un *point* et contenir uniquement des *lettres*, des *chiffres*, des *soulignements*, des *Hyphens* et des *points*. Les caractères valides sont définis dans la spécification XML W3C et consistent essentiellement en des catégories Unicode L1, Lu, Lo, Lt, Nl, Mc, Mn, Lm, et Nd ; voir [Catégories de caractères Unicode](#).

Le chemin complet d'une propriété ou d'une paire nom-valeur ne doit pas enfreindre la règle décrite dans «[Nom de chemin d'accès non valide](#)», à la [page 572](#). Les chemins d'accès sont limités à 4095 octets, ne doivent pas contenir de caractères de compatibilité Unicode et ne doivent pas commencer par la chaîne XML.

## Nom du groupe

Un nom de groupe a la même syntaxe qu'un nom. Les noms de groupe sont facultatifs. Les propriétés et les paires nom-valeur peuvent être placées à la racine d'un dossier. Utilisez des groupes si cela permet d'organiser les propriétés et les paires nom-valeur.

## Nom de l'élément

Un nom d'élément a la même syntaxe qu'un nom.

## Valeur d'élément

Une valeur d'élément inclut tous les espaces entre la balise `< Element name >` et `< /Element name >`. N'utilisez pas les deux caractères `<` et `&` dans une valeur. Remplacez ensuite par `<` et `&` ;.

## Attributs de propriété

Les attributs de propriété mappent les zones de descripteur de propriété: Les mappages sont les suivants:

## Support

### sa (valeur par défaut)

MQPD\_SUPPORT\_OPTIONAL

### sr

MQPD\_SUPPORT\_REQUIRED

### sx

MQPD\_SUPPORT\_REQUIRED\_IF\_LOCAL

## Contexte

### none (par défaut)

MQPD\_NO\_CONTEXT

### utilisateur

MQPD\_USER \_CONTEXT

## CopyOptions

### réacheminement

MQPD\_COPY\_FORWARD

### réponse

MQPD\_COPY\_REPLY

### rapport

MQPD\_COPY\_REPORT

### publier

MQPD\_COPY\_PUBLISH

### tout

MQPD\_COPY\_ALL

N'utilisez pas all en combinaison avec d'autres options.

### par défaut

MQPD\_COPY\_DEFAULT

N'utilisez pas default en combinaison avec d'autres options. default est identique à forward + report + publish.

### Aucun

MQPD\_COPY\_NONE

N'utilisez pas none avec d'autres options.

Les attributs de la propriété Support ne s'appliquent qu'aux propriétés du dossier mq .

Les attributs de propriété Context et CopyOptions sont applicables à tous les dossiers de propriétés.

## types de données

Les types de données MQRFH2 sont mappés aux types de propriété de message comme suit:

Type de données MQRFH2	Type de propriété de message
bin.hex	MQBYTE []
boolean	MQBOOL
i1	MQINT8
i2	MQINT16
i4	MQINT32

Tableau 522. Mappages de type de données (suite)

Type de données MQRFH2	Type de propriété de message
i8	MQINT64
r4	MQFLOAT32
r8	MQFLOAT64
string	MQCHAR[]

Tout élément sans type de données est supposé être de type string.

Une valeur nulle est indiquée par l'attribut d'élément `xsi:nil='true'`. N'utilisez pas l'attribut `xsi:nil='false'` pour les valeurs non nulles. Par exemple, la propriété suivante a une valeur null:

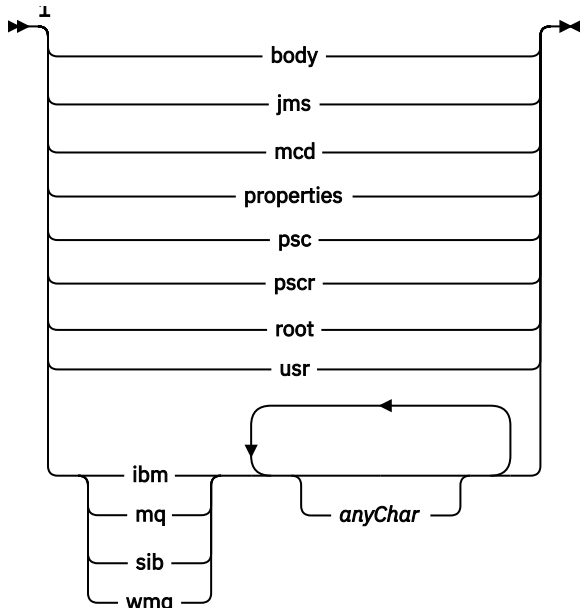
```
<NullProperty
xsi:nil='true'></NullProperty>
```

Une propriété d'octet ou de chaîne de caractères peut avoir une valeur vide. Une valeur vide est représentée par un élément MQRFH2 avec une valeur d'élément de longueur nulle. Par exemple, la propriété suivante a une valeur vide:

```
<EmptyProperty></EmptyProperty>
```

### Nom du dossier réservé ou du dossier de propriétés

Limitez le nom d'un dossier ou d'un dossier de propriétés à ne pas commencer par l'une des chaînes suivantes. Les préfixes sont réservés aux noms de dossier ou de propriété créés par IBM.

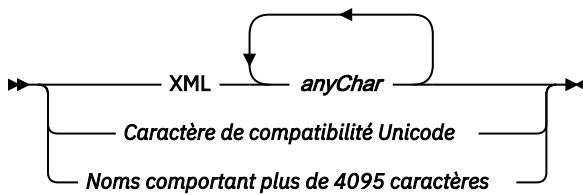


Remarques :

- <sup>1</sup> Un nom de dossier ou de propriété réservé contient une combinaison de lettres minuscules et majuscules.

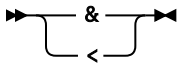
### Nom de chemin d'accès non valide

Limitez le chemin complet d'une paire nom-valeur ou d'une propriété pour ne pas inclure l'une des chaînes suivantes.



## Caractères non valides

Utilisez toujours les séquences d'échappement & ; et < à la place des littéraux "&" et "<".

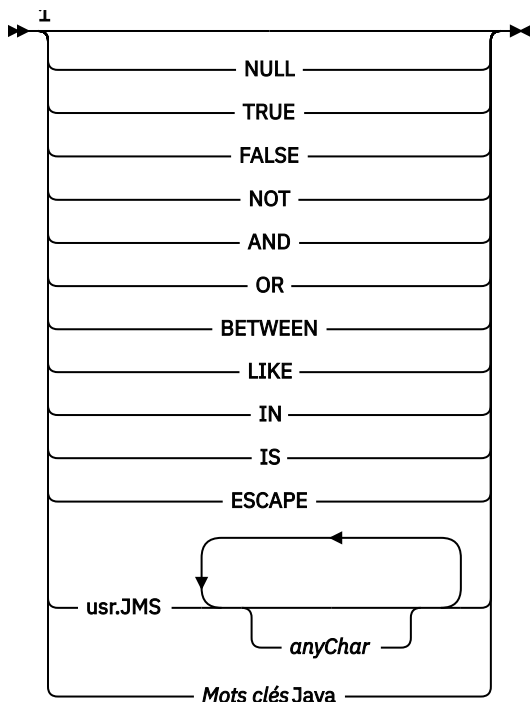


## Noms de propriété définis

Les noms de propriété définis sont les noms des propriétés définies par IBM MQ ou d'autres produits et utilisées par IBM MQ et les applications utilisateur. Les propriétés définies existent uniquement dans les dossiers de propriétés définis. Les noms de propriété définis sont décrits dans la description des dossiers de propriétés ; voir [Dossiers de propriétés](#).

## Nom de propriété non valide

Ne construisez pas de noms de propriété qui correspondent à la règle suivante. La règle s'applique au chemin d'accès complet à la propriété qui nomme une propriété, et pas seulement au nom de l'élément de propriété.



Remarques :

<sup>1</sup> Un nom de propriété non valide peut contenir n'importe quelle combinaison de majuscules et de minuscules.

## Attributs non valides

Les dossiers de propriétés et les propriétés ne peuvent inclure que les éléments «Attributs de propriété», à la page 570 et «types de données», à la page 571 pris en charge.

Tous les attributs de type XML non pris en charge, par exemple les noms avec des valeurs de chaîne entre guillemets, qui sont inclus dans les dossiers de propriétés ou les propriétés, peuvent être supprimés.

Attributs de type XML inclus dans les dossiers non-propriété ou les éléments non-propriété qui restent dans les en-têtes MQRFH2 .

## MQRMH-En-tête de message de référence

La structure MQRMH définit le format d'un en-tête de message de référence. Cet en-tête est utilisé avec les exits de canal de message écrits par l'utilisateur pour envoyer de très grandes quantités de données (appelées *données en masse*) d'un gestionnaire de files d'attente à un autre. La différence par rapport à la messagerie normale est que les données non formatées ne sont pas stockées dans une file d'attente ; à la place, seule une *référence* aux données non formatées est stockée dans la file d'attente. Cela réduit la possibilité que les ressources IBM MQ soient épuisées par un petit nombre de messages extrêmement volumineux.

## Disponibilité

La structure MQRMH est disponible sur les plateformes suivantes:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour les clients IBM MQ connectés à ces systèmes.

## Format

MQFMT\_REF\_MSG\_HEADER

## Jeu de caractères et codage

Les données de type caractère dans MQRMH et les chaînes traitées par les zones de décalage doivent figurer dans le jeu de caractères du gestionnaire de files d'attente local ; cet attribut est fourni par l'attribut de gestionnaire de files d'attente **CodedCharSetId** . Les données numériques dans MQRMH doivent être dans le codage machine natif ; elles sont fournies par la valeur de MQENC\_NATIVE pour le langage de programmation C.

Définissez le jeu de caractères et le codage du MQRMH dans les zones *CodedCharSetId* et *Encoding* dans:

- MQMD (si la structure MQRMH est au début des données de message), ou
- Structure d'en-tête qui précède la structure MQRMH (tous les autres cas).

## Utilisation

Une application insère un message composé d'un MQRMH, mais en omettant les données non formatées. Lorsqu'un agent MCA lit le message à partir de la file d'attente de transmission, un exit de message fourni par l'utilisateur est appelé pour traiter l'en-tête de message de référence. L'exit peut ajouter au message de référence les données non formatées identifiées par la structure MQRMH, avant que l'agent MCA n'envoie le message via le canal au gestionnaire de files d'attente suivant.

A l'extrémité réceptrice, un exit de message qui attend les messages de référence doit exister. Lorsqu'un message de référence est reçu, l'exit doit créer l'objet à partir des données non formatées qui suivent le MQRMH dans le message, puis transmettre le message de référence sans les données non formatées. Le message de référence peut ensuite être extrait par une application lisant le message de référence (sans les données non formatées) à partir d'une file d'attente.

Normalement, la structure MQRMH est tout ce qui se trouve dans le message. Toutefois, si le message se trouve dans une file d'attente de transmission, un ou plusieurs en-têtes supplémentaires précèdent la structure MQRMH.

Un message de référence peut également être envoyé à une liste de distribution. Dans ce cas, la structure MQDH et ses enregistrements associés précèdent la structure MQRMH lorsque le message se trouve dans une file d'attente de transmission.

**Remarque :** N'envoyez pas de message de référence en tant que message segmenté, car l'exit de message ne peut pas le traiter correctement.

## Conversion de données

A des fins de conversion de données, la conversion de la structure MQRMH inclut la conversion des données d'environnement source, du nom d'objet source, des données d'environnement de destination et du nom d'objet de destination. Tous les autres octets compris dans *StrucLength* octets du début de la structure sont supprimés ou ont des valeurs non définies après la conversion des données. Les données non formatées sont converties à condition que toutes les instructions suivantes soient vraies:

- Les données non formatées sont présentes dans le message lorsque la conversion de données est effectuée.
- La valeur de la zone *Format* dans MQRMH est différente de MQFMT\_NONE.
- Un exit de conversion de données écrit par l'utilisateur existe avec le nom de format indiqué.

Sachez toutefois que les données non formatées ne sont généralement pas présentes dans le message lorsque le message se trouve dans une file d'attente et que, par conséquent, les données non formatées sont converties par l'option MQGMO\_CONVERT.

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Tableau 523. Zones de MQRMH pour MQRMH		
Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID_STRUC_MQRMH_	'RMH→'
<u>Version</u> (numéro de version de la structure)	MQRMH_VERSION_1	1
<u>StrucLength</u> (longueur totale de MQRMH, y compris les chaînes à la fin des zones fixes, mais pas les données non formatées)	Aucun	0
<u>Codage</u> (codage numérique des données non formatées)	MQENC_NATIVE	Dépend de l'environnement
<u>CodedCharSetId</u> (identificateur de jeu de caractères des données non formatées)	MQCCSI_UNDEFINI	0
<u>Format</u> (nom de format des données non formatées)	MQFMT_AUCUN	Espaces vides
<u>Indicateurs</u> (indicateurs de message de référence)	MQRMHF_NOT_LAST	0

Tableau 523. Zones de MQRMH pour MQRMH (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>ObjectType</u> (type d'objet)	Aucun	Espaces vides
<u>ObjectInstanceId</u> (identificateur de l'instance d'objet)	MQOII_AUCUN	Valeurs NULL
<u>SrcEnvLongueur</u> (longueur des données d'environnement source)	Aucun	0
<u>DécalageSrcEnv</u> (décalage des données d'environnement source)	Aucun	0
<u>SrcNameLongueur</u> (longueur du nom d'objet source)	Aucun	0
<u>SrcNameDécalage</u> (décalage du nom de l'objet source)	Aucun	0
<u>DestEnvLongueur</u> (longueur des données d'environnement de destination)	Aucun	0
<u>DestEnvDécalage</u> (décalage des données d'environnement de destination)	Aucun	0
<u>DestNameLongueur</u> (longueur du nom d'objet de destination)	Aucun	0
<u>DestNameDécalage</u> (décalage du nom d'objet de destination)	Aucun	0
<u>DataLogicalLongueur</u> (longueur des données non formatées)	Aucun	0
<u>DataLogicalOffset</u> (décalage faible des données non formatées)	Aucun	0
<u>DataLogicalOffset2</u> (décalage élevé des données non formatées)	Aucun	0

**Remarques :**

1. Le symbole ↵ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable macro MQRMH\_DEFAULT contient les valeurs répertoriées dans la table. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQRMH MyRMH = {MQRMH_DEFAULT};
```

## Déclarations de langue

Déclaration C pour MQRMH

```
typedef struct tagMQRMH MQRMH;
struct tagMQRMH {
    MQCHAR4   StructId;           /* Structure identifier */
    MQLONG    Version;           /* Structure version number */
    MQLONG    StruLength;        /* Total length of MQRMH, including
                                strings at end of fixed fields, but
```



```

                                not the bulk data */
MQLONG   Encoding;                /* Numeric encoding of bulk data */
MQLONG   CodedCharSetId;         /* Character set identifier of bulk
                                data */
MQCHAR8  Format;                  /* Format name of bulk data */
MQLONG   Flags;                  /* Reference message flags */
MQCHAR8  ObjectType;             /* Object type */
MQBYTE24 ObjectInstanceId;       /* Object instance identifier */
MQLONG   SrcEnvLength;           /* Length of source environment data */
MQLONG   SrcEnvOffset;           /* Offset of source environment data */
MQLONG   SrcNameLength;         /* Length of source object name */
MQLONG   SrcNameOffset;         /* Offset of source object name */
MQLONG   DestEnvLength;         /* Length of destination environment
                                data */
MQLONG   DestEnvOffset;         /* Offset of destination environment
                                data */
MQLONG   DestNameLength;        /* Length of destination object name */
MQLONG   DestNameOffset;        /* Offset of destination object name */
MQLONG   DataLogicalLength;     /* Length of bulk data */
MQLONG   DataLogicalOffset;     /* Low offset of bulk data */
MQLONG   DataLogicalOffset2;    /* High offset of bulk data */
};

```

### Déclaration COBOL pour MQRMH

```

** MQRMH structure
10 MQRMH.
** Structure identifier
15 MQRMH-STRUCID PIC X(4).
** Structure version number
15 MQRMH-VERSION PIC S9(9) BINARY.
** Total length of MQRMH, including strings at end of fixed fields,
** but not the bulk data
15 MQRMH-STRUCLength PIC S9(9) BINARY.
** Numeric encoding of bulk data
15 MQRMH-ENCODING PIC S9(9) BINARY.
** Character set identifier of bulk data
15 MQRMH-CODEDCHARSETID PIC S9(9) BINARY.
** Format name of bulk data
15 MQRMH-FORMAT PIC X(8).
** Reference message flags
15 MQRMH-FLAGS PIC S9(9) BINARY.
** Object type
15 MQRMH-OBJECTTYPE PIC X(8).
** Object instance identifier
15 MQRMH-OBJECTINSTANCEID PIC X(24).
** Length of source environment data
15 MQRMH-SRCENVLENGTH PIC S9(9) BINARY.
** Offset of source environment data
15 MQRMH-SRCENVOFFSET PIC S9(9) BINARY.
** Length of source object name
15 MQRMH-SRCNAMELENGTH PIC S9(9) BINARY.
** Offset of source object name
15 MQRMH-SRCNAMEOFFSET PIC S9(9) BINARY.
** Length of destination environment data
15 MQRMH-DESTENVLENGTH PIC S9(9) BINARY.
** Offset of destination environment data
15 MQRMH-DESTENVOFFSET PIC S9(9) BINARY.
** Length of destination object name
15 MQRMH-DESTNAMELENGTH PIC S9(9) BINARY.
** Offset of destination object name
15 MQRMH-DESTNAMEOFFSET PIC S9(9) BINARY.
** Length of bulk data
15 MQRMH-DATALOGICALENGTH PIC S9(9) BINARY.
** Low offset of bulk data
15 MQRMH-DATALOGICALOFFSET PIC S9(9) BINARY.
** High offset of bulk data
15 MQRMH-DATALOGICALOFFSET2 PIC S9(9) BINARY.

```

### Déclaration PL/I pour MQRMH

```

dcl
  1 MQRMH based,
  3 StrucId char(4), /* Structure identifier */
  3 Version fixed bin(31), /* Structure version number */
  3 StrucLength fixed bin(31), /* Total length of MQRMH,
                                including strings at end of

```

		fixed fields, but not the bulk data */
3 Encoding	fixed bin(31), /*	Numeric encoding of bulk data */
3 CodedCharSetId	fixed bin(31), /*	Character set identifier of bulk data */
3 Format	char(8), /*	Format name of bulk data */
3 Flags	fixed bin(31), /*	Reference message flags */
3 ObjectType	char(8), /*	Object type */
3 ObjectInstanceId	char(24), /*	Object instance identifier */
3 SrcEnvLength	fixed bin(31), /*	Length of source environment data */
3 SrcEnvOffset	fixed bin(31), /*	Offset of source environment data */
3 SrcNameLength	fixed bin(31), /*	Length of source object name */
3 SrcNameOffset	fixed bin(31), /*	Offset of source object name */
3 DestEnvLength	fixed bin(31), /*	Length of destination environment data */
3 DestEnvOffset	fixed bin(31), /*	Offset of destination environment data */
3 DestNameLength	fixed bin(31), /*	Length of destination object name */
3 DestNameOffset	fixed bin(31), /*	Offset of destination object name */
3 DataLogicalLength	fixed bin(31), /*	Length of bulk data */
3 DataLogicalOffset	fixed bin(31), /*	Low offset of bulk data */
3 DataLogicalOffset2	fixed bin(31); /*	High offset of bulk data */

### Déclaration High Level Assembler pour MQRMH

```

MQRMH          DSECT
MQRMH_STRUCID  DS   CL4   Structure identifier
MQRMH_VERSION  DS   F     Structure version number
MQRMH_STRUCLNGTH DS   F     Total length of MQRMH, including
*               strings at end of fixed fields, but
*               not the bulk data
MQRMH_ENCODING DS   F     Numeric encoding of bulk data
MQRMH_CODEDCHARSETID DS   F   Character set identifier of bulk
*               data
MQRMH_FORMAT   DS   CL8   Format name of bulk data
MQRMH_FLAGS    DS   F     Reference message flags
MQRMH_OBJECTTYPE DS   CL8  Object type
MQRMH_OBJECTINSTANCEID DS XL24 Object instance identifier
MQRMH_SRCENVLENGTH DS   F   Length of source environment data
MQRMH_SRCENVOFFSET DS   F   Offset of source environment data
MQRMH_SRCNAMELENGTH DS   F   Length of source object name
MQRMH_SRCNAMEOFFSET DS   F   Offset of source object name
MQRMH_DESTENVLENGTH DS   F   Length of destination environment
*               data
MQRMH_DESTENVOFFSET DS   F   Offset of destination environment
*               data
MQRMH_DESTNAMELENGTH DS   F   Length of destination object name
MQRMH_DESTNAMEOFFSET DS   F   Offset of destination object name
MQRMH_DATALOGICALLLENGTH DS   F   Length of bulk data
MQRMH_DATALOGICALOFFSET DS   F   Low offset of bulk data
MQRMH_DATALOGICALOFFSET2 DS   F   High offset of bulk data
*
MQRMH_LENGTH   EQU   *-MQRMH
MQRMH_AREA     DS     CL(MQRMH_LENGTH)

```

### Déclaration Visual Basic pour MQRMH

```

Type MQRMH
  StrucId      As String*4 'Structure identifier'
  Version      As Long     'Structure version number'
  StrucLength  As Long     'Total length of MQRMH, including
  'strings at end of fixed fields, but'
  'not the bulk data'
  Encoding     As Long     'Numeric encoding of bulk data'
  CodedCharSetId As Long   'Character set identifier of bulk data'
  Format       As String*8 'Format name of bulk data'
  Flags        As Long     'Reference message flags'
  ObjectType   As String*8 'Object type'
  ObjectInstanceId As MQBYTE24 'Object instance identifier'
  SrcEnvLength As Long     'Length of source environment data'
  SrcEnvOffset As Long     'Offset of source environment data'

```

SrcNameLength	As Long	'Length of source object name'
SrcNameOffset	As Long	'Offset of source object name'
DestEnvLength	As Long	'Length of destination environment 'data'
DestEnvOffset	As Long	'Offset of destination environment 'data'
DestNameLength	As Long	'Length of destination object name'
DestNameOffset	As Long	'Offset of destination object name'
DataLogicalLength	As Long	'Length of bulk data'
DataLogicalOffset	As Long	'Low offset of bulk data'
DataLogicalOffset2	As Long	'High offset of bulk data'
End Type		

### ***StrucId (MCHAR4) pour MQRMH***

Il s'agit de l'identificateur de structure de la structure d'en-tête de message de référence. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQRMH\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUC\_MQRMH\_**

Identificateur de la structure d'en-tête de message de référence.

Pour le langage de programmation C, la constante MQRMH\_STRUC\_ID\_ARRAY est également définie. Cette valeur est identique à MQRMH\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### ***Version (MQLONG) pour MQRMH***

Numéro de version de la structure. La valeur doit être:

#### **MQRMH\_VERSION\_1**

La structure d'en-tête de message de référence Version-1 .

La constante suivante indique le numéro de version de la version en cours:

#### **MQRMH\_VERSION\_CURRENT\_VERSION**

Version actuelle de la structure d'en-tête de message de référence.

La valeur initiale de cette zone est MQRMH\_VERSION\_1.

### ***StrucLength (MQLONG) pour MQRMH***

Longueur totale de MQRMH, y compris les chaînes à la fin des zones fixes, mais pas les données non formatées.

La valeur initiale de cette zone est zéro.

### ***Codage (MQLONG) pour MQRMH***

Indique le codage numérique des données non formatées ; il ne s'applique pas aux données numériques de la structure MQRMH elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données.

La valeur initiale de cette zone est MQENC\_NATIVE.

### ***CodedCharSetId (MQLONG) pour MQRMH***

Indique l'identificateur de jeu de caractères des données non formatées ; il ne s'applique pas aux données de type caractères de la structure MQRMH elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. La valeur spéciale suivante peut être utilisée:

#### **MQCCSI\_HÉRITER**

Les données de type caractères des données qui suivent cette structure sont dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, la valeur MQCCSI\_INHERIT n'est pas renvoyée par l'appel MQGET.

N'utilisez pas MQCCSI\_INHERIT si la valeur de la zone PutApp1Type dans MQMD est MQAT\_BROKER.

Cette valeur est prise en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour les clients IBM MQ connectés à ces systèmes.

La valeur initiale de cette zone est MQCCSI\_UNDEFINED.

### **Format (MQCHAR8) pour MQRMH**

Indique le nom de format des données non formatées.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de la zone *Format* dans MQMD.

La valeur initiale de cette zone est MQFMT\_NONE.

### **Indicateurs (MQLONG) pour MQRMH**

Il s'agit d'indicateurs de message de référence. Les indicateurs suivants sont définis:

#### **MQRMHF\_LAST**

Cet indicateur indique que le message de référence représente ou contient la dernière partie de l'objet référencé.

#### **MQRMHF\_NOT\_LAST**

Le message de référence ne contient pas ou ne représente pas la dernière partie de l'objet.

MQRMHF\_NOT\_LAST aide la documentation du programme. Il n'est pas prévu que cette option soit utilisée avec une autre option, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

La valeur initiale de cette zone est MQRMHF\_NOT\_LAST.

### **ObjectType (MQCHAR8) pour MQRMH**

Il s'agit d'un nom que l'exit de message peut utiliser pour reconnaître les types de message de référence qu'il prend en charge. Le nom doit être conforme aux mêmes règles que la zone *Format*. Voir [«Format \(MQCHAR8\) pour MQRMH»](#), à la page 580.

La valeur initiale de cette zone est 8 blancs.

### **ID ObjectInstance(MQBYTE24) pour MQRMH**

Utilisez cette zone pour identifier une instance spécifique d'un objet. S'il n'est pas nécessaire, définissez-le sur la valeur suivante:

#### **MQOII\_AUCUN**

Aucun identificateur d'instance d'objet spécifié. La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQOII\_NONE\_ARRAY est également définie ; elle a la même valeur que MQOII\_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La longueur de cette zone est indiquée par MQ\_OBJECT\_INSTANCE\_ID\_LENGTH. La valeur initiale de cette zone est MQOII\_NONE.

### ***SrcEnvLongueur (MQLONG) pour MQRMH***

Longueur des données d'environnement source. Si cette zone est définie sur zéro, il n'y a pas de données d'environnement source et *SrcEnvOffset* est ignoré.

La valeur initiale de cette zone est 0.

### ***SrcEnvDécalage (MQLONG)***

Cette zone indique le décalage des données d'environnement source à partir du début de la structure MQRMH. Les données d'environnement source peuvent être spécifiées par le créateur du message de référence, si ces données sont connues du créateur. Par exemple, sous Windows, les données d'environnement source peuvent être le chemin de répertoire de l'objet contenant les données non formatées. Toutefois, si le créateur ne connaît pas les données de l'environnement source, l'exit de message fourni par l'utilisateur doit déterminer les informations d'environnement requises.

La longueur des données d'environnement source est indiquée par *SrcEnvLength* ; Si cette longueur est égale à zéro, il n'y a pas de données d'environnement source et *SrcEnvOffset* est ignoré. Si elles sont présentes, les données de l'environnement source doivent résider entièrement dans *StrucLength* octets à partir du début de la structure.

Les applications ne doivent pas supposer que les données d'environnement démarrent immédiatement après la dernière zone fixe de la structure ou qu'elles sont contiguës à l'une des données traitées par les zones *SrcNameOffset*, *DestEnvOffset* et *DestNameOffset*.

La valeur initiale de cette zone est 0.

### ***SrcNameLongueur (MQLONG) pour MQRMH***

Longueur du nom de l'objet source. Si cette zone est à zéro, il n'y a pas de nom d'objet source et *SrcNameOffset* est ignoré.

La valeur initiale de cette zone est 0.

### ***SrcNameDécalage (MQLONG) pour MQRMH***

Cette zone indique le décalage du nom d'objet source à partir du début de la structure MQRMH. Le nom de l'objet source peut être spécifié par le créateur du message de référence, si ces données sont connues du créateur. Toutefois, si le créateur ne connaît pas le nom de l'objet source, l'exit de message fourni par l'utilisateur doit identifier l'objet à accéder.

La longueur du nom d'objet source est donnée par *SrcNameLength* ; Si cette longueur est égale à zéro, il n'y a pas de nom d'objet source et *SrcNameOffset* est ignoré. S'il est présent, le nom de l'objet source doit se trouver complètement dans *StrucLength* octets à partir du début de la structure.

Les applications ne doivent pas supposer que le nom de l'objet source est contigu à l'une des données traitées par les zones *SrcEnvOffset*, *DestEnvOffset* et *DestNameOffset*.

La valeur initiale de cette zone est 0.

### ***DestEnvLongueur (MQLONG) pour MQRMH***

Longueur des données de l'environnement de destination. Si cette zone a pour valeur zéro, il n'y a pas de données d'environnement de destination et *DestEnvOffset* est ignoré.

### ***DestEnvDécalage (MQLONG) pour MQRMH***

Cette zone indique le décalage des données d'environnement de destination à partir du début de la structure MQRMH. Les données d'environnement de destination peuvent être spécifiées par le créateur du message de référence, si ces données sont connues du créateur. Par exemple, sous Windows, les données de l'environnement de destination peuvent être le chemin de répertoire de l'objet dans lequel les données non formatées doivent être stockées. Toutefois, si le créateur ne connaît pas les données d'environnement de destination, il incombe à l'exit de message fourni par l'utilisateur de déterminer les informations d'environnement requises.

La longueur des données de l'environnement de destination est indiquée par *DestEnvLength* ; Si cette longueur est égale à zéro, il n'y a pas de données d'environnement de destination et *DestEnvOffset* est ignoré. S'il est présent, les données de l'environnement de destination doivent résider entièrement dans *StrucLength* octets à partir du début de la structure.

Les applications ne doivent pas supposer que les données de l'environnement de destination sont contiguës à l'une des données traitées par les zones *SrcEnvOffset*, *SrcNameOffset* et *DestNameOffset*.

La valeur initiale de cette zone est 0.

### ***DestNameLongueur (MQLONG) pour MQRMH***

Longueur du nom de l'objet de destination. Si cette zone est à zéro, il n'y a pas de nom d'objet de destination et *DestNameOffset* est ignoré.

### ***DestNameDécalage (MQLONG) pour MQRMH***

Cette zone indique le décalage du nom d'objet de destination à partir du début de la structure MQRMH. Le nom de l'objet de destination peut être spécifié par le créateur du message de référence, si ces données sont connues du créateur. Toutefois, si le créateur ne connaît pas le nom de l'objet de destination, il incombe à l'exit de message fourni par l'utilisateur d'identifier l'objet à créer ou à modifier.

La longueur du nom de l'objet de destination est donnée par *DestNameLength* ; Si cette longueur est égale à zéro, il n'y a pas de nom d'objet de destination et *DestNameOffset* est ignoré. S'il est présent, le nom de l'objet de destination doit résider entièrement dans *StrucLength* octets à partir du début de la structure.

Les applications ne doivent pas supposer que le nom de l'objet de destination est contigu à l'une des données traitées par les zones *SrcEnvOffset*, *SrcNameOffset* et *DestEnvOffset*.

La valeur initiale de cette zone est 0.

### ***DataLogicallogique de données (MQLONG) pour MQRMH***

La zone *DataLogicalLength* indique la longueur des données non formatées référencées par la structure MQRMH.

Si les données non formatées sont effectivement présentes dans le message, les données commencent à un décalage de *StrucLength* octets à partir du début de la structure MQRMH. La longueur de l'ensemble du message moins *StrucLength* indique la longueur des données non formatées présentes.

Si des données sont présentes dans le message, *DataLogicalLength* indique la quantité de données pertinentes. Dans le cas normal, *DataLogicalLength* doit avoir la même valeur que la longueur des données présentes dans le message.

Si la structure MQRMH représente les données restantes dans l'objet (à partir du décalage logique spécifié), vous pouvez utiliser la valeur zéro pour *DataLogicalLength*, à condition que les données non formatées ne soient pas réellement présentes dans le message.

Si aucune donnée n'est présente, la fin de MQRMH coïncide avec la fin du message.

La valeur initiale de cette zone est 0.

### ***DataLogicalOffset (MQLONG) pour MQRMH***

Cette zone indique le décalage faible des données non formatées à partir du début de l'objet dont font partie les données non formatées. Le décalage des données non formatées à partir du début de l'objet est appelé *décalage logique*. Il ne s'agit pas du décalage physique des données non formatées à partir du début de la structure MQRMH ; ce décalage est fourni par *StrucLength*.

Pour permettre l'envoi d'objets volumineux à l'aide de messages de référence, le décalage logique est divisé en deux zones et le décalage logique réel est donné par la somme de ces deux zones:

- *DataLogicalOffset* représente le reste obtenu lorsque le décalage logique est divisé par 1 000 000 000. Il s'agit donc d'une valeur comprise entre 0 et 999 999 999.
- *DataLogicalOffset2* représente le résultat obtenu lorsque le décalage logique est divisé par 1 000 000 000. C'est donc le nombre de multiples complets de 1 000 000 000 qui existent dans le décalage logique. Le nombre de multiples est compris entre 0 et 999 999 999.

La valeur initiale de cette zone est 0.

### **DataLogicalOffset2 (MQLONG) pour MQRMH**

Cette zone indique le décalage élevé des données non formatées à partir du début de l'objet dont font partie les données non formatées. Il s'agit d'une valeur comprise entre 0 et 999 999 999. Voir *DataLogicalOffset* pour des détails.

La valeur initiale de cette zone est 0.

## **MQRR-Enregistrement de réponse**

Utilisez la structure MQRR pour recevoir le code achèvement et le code anomalie résultant de l'opération d'ouverture ou d'insertion pour une file d'attente de destination unique, lorsque la destination est une liste de distribution. MQRR est une structure de sortie pour les appels MQOPEN, MQPUT et MQPUT1 .

### **Disponibilité**

La structure MQRR est disponible sur les plateformes suivantes:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour les clients IBM MQ connectés à ces systèmes.

### **Jeu de caractères et codage**

Les données de MQRR doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par MQENC\_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

### **Utilisation**

En fournissant un tableau de ces structures sur les appels MQOPEN et MQPUT ou sur l'appel MQPUT1 , vous pouvez déterminer les codes achèvement et les codes anomalie de toutes les files d'attente d'une liste de distribution lorsque le résultat de l'appel est mixte, c'est-à-dire lorsque l'appel aboutit pour certaines files d'attente de la liste mais échoue pour d'autres. Le code anomalie MQRC\_MULTIPLE\_MOTIFS de l'appel indique que les enregistrements de réponse (s'ils sont fournis par l'application) ont été définis par le gestionnaire de files d'attente.

### **Zones**

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Tableau 524. Zones dans MQRR

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
CompCode (code achèvement de la file d'attente)	MQCC_OK	0
Raison (code raison de la file d'attente)	MQRC_NONE	0
<p><b>Remarques :</b></p> <p>1. Dans le langage de programmation C, la variable macroMQRR_DEFAULT contient les valeurs répertoriées dans la table. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</p> <pre style="background-color: #f0f0f0; padding: 5px;">MQRR MyRR = {MQRR_DEFAULT};</pre>		

## Déclarations de langue

### Déclaration C pour MQRR

```
typedef struct tagMQRR MQRR;
struct tagMQRR {
    MQLONG CompCode; /* Completion code for queue */
    MQLONG Reason; /* Reason code for queue */
};
```

### Déclaration COBOL pour MQRR

```
** MQRR structure
10 MQRR.
** Completion code for queue
15 MQRR-COMPCODE PIC S9(9) BINARY.
** Reason code for queue
15 MQRR-REASON PIC S9(9) BINARY.
```

### Déclaration PL/I pour MQRR

```
dcl
1 MQRR based,
3 CompCode fixed bin(31), /* Completion code for queue */
3 Reason fixed bin(31); /* Reason code for queue */
```

### Déclaration Visual Basic pour MQRR

```
Type MQRR
CompCode As Long 'Completion code for queue'
Reason As Long 'Reason code for queue'
End Type
```

## CompCode (MQLONG) pour MQRR

Il s'agit du code achèvement résultant de l'opération d'ouverture ou d'insertion pour la file d'attente avec le nom indiqué par l'élément correspondant dans le tableau des structures MQOR fournies sur l'appel MQOPEN ou MQPUT1 .

Il s'agit toujours d'une zone de sortie. La valeur initiale de cette zone est MQCC\_OK.

## Motif (MQLONG) pour MQRR



Il s'agit du code anomalie résultant de l'opération d'ouverture ou d'insertion pour la file d'attente avec le nom qui a été indiqué par l'élément correspondant dans le tableau des structures MQOR fournies sur l'appel MQOPEN ou MQPUT1 .




Il s'agit toujours d'une zone de sortie. La valeur initiale de cette zone est MQRC\_NONE.

## MQSCO-Options de configuration SSL/TLS

La structure MQSCO, associée aux zones TLS de la structure MQCD, permet à une application s'exécutant en tant que IBM MQ MQI client de spécifier des options de configuration qui contrôlent l'utilisation de TLS pour la connexion client lorsque le protocole de canal est TCP/IP. La structure est un paramètre d'entrée dans l'appel MQCONN.

### Disponibilité

La structure MQSCO est disponible sur les clients suivants:

-  AIX
-  IBM i
-  Linux
-  Windows

Si le protocole de canal du canal client n'est pas TCP/IP, la structure MQSCO est ignorée.

### Jeu de caractères et codage

Les données de MQSCO doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par MQENC\_NATIVE.




### Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.


*Tableau 525. Champs dans MQSCO*

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	MQSCO_STRUC_ID	'SCO↵'
<u>Version</u> (numéro de version de la structure)	MQSCO_CURRENT_VERSION	1
<u>KeyRepository</u> (emplacement du référentiel de clés)	Aucun	Chaîne nulle ou blancs
<u>CryptoHardware</u> (détails du matériel de cryptographie)	Aucun	Chaîne nulle ou blancs
<u>AuthInfoRecCount</u> (nombre d'enregistrements MQAIR présents)	Aucun	0
<u>AuthInfoRecOffset</u> (décalage du premier enregistrement MQAIR à partir du début de MQSCO)	Aucun	0

Tableau 525. Champs dans MQSCO (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
AuthInfoRecPtr (adresse du premier enregistrement MQAIR)	Aucun	Pointeur null ou octets null
<b>Remarque :</b> Les deux zones suivantes sont ignorées si <i>Version</i> est inférieur à MQSCO_VERSION_2.		
KeyResetNombre (nombre de réinitialisations de clé confidentielle TLS)	MQSCO_RESET_COUNT_DEFAULT	0
«FipsRequired (MQLONG) pour MQSCO», à la page 591 (utilisez des algorithmes de cryptographie certifiés FIPS dans IBM MQ)	MQSSL_FIPS_NO	0
<b>Remarque :</b> Les deux zones suivantes sont ignorées si <i>Version</i> est inférieur à MQSCO_VERSION_3.		
EncryptionPolicySuiteB (utiliser uniquement les algorithmes de cryptographie Suite B)	MQ_SUITE_B_NONE, MQ_SUITE_B_NOT_AVAILABLE, MQ_SUITE_B_NOT_AVAILABLE, MQ_SUITE_B_NOT_AVAILABLE	1, 0, 0, 0
<b>Remarque :</b> Les deux zones suivantes sont ignorées si <i>Version</i> est inférieur à MQSCO_VERSION_4.		
PolitiqueCertificateVal (politique de validation de certificat)	MQ_CERT_VAL_POLICY_DEFAULT	0
<b>Remarque :</b> Les deux zones suivantes sont ignorées si <i>Version</i> est inférieur à MQSCO_VERSION_5.		
CertificateLabel (indique le libellé de certificat utilisé)	Aucun	Chaîne nulle ou blancs
<b>Remarque :</b> Les zones restantes sont ignorées si <i>Version</i> est inférieur à MQSCO_VERSION_6.		
 KeyRepoPasswordPtr (adresse du mot de passe du référentiel de clés TLS)	Aucun	Pointeur null ou octets null
 KeyRepoPasswordOffset (décalage du mot de passe du référentiel de clés TLS)	Aucun	0
 KeyRepoPasswordLength (longueur du mot de passe du référentiel de clés TLS)	Aucun	0

**Remarques :**

1. Le symbole  représente un caractère blanc unique.

2. Dans le langage de programmation C, la variable macro MQSCO\_DEFAULT contient les valeurs répertoriées dans le tableau. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQSCO MySCO = {MQSCO_DEFAULT};
```

## Déclarations de langue

### Déclaration C pour MQSCO

```
typedef struct tagMQSCO MQSCO;
struct tagMQSCO {
    MQCHAR4    StrucId;                /* Structure identifier */
    MQLONG     Version;                /* Structure version number */
    MQCHAR256  KeyRepository;          /* Location of TLS key */
                                          /* repository */
    MQCHAR256  CryptoHardware;         /* Cryptographic hardware */
                                          /* configuration string */
    MQLONG     AuthInfoRecCount;       /* Number of MQAIR records */
                                          /* present */
    MQLONG     AuthInfoRecOffset;      /* Offset of first MQAIR */
                                          /* record from start of */
                                          /* MQSCO structure */
    PMQAIR     AuthInfoRecPtr;         /* Address of first MQAIR */
                                          /* record */
    /* Ver:1 */
    MQLONG     KeyResetCount;          /* Number of unencrypted */
                                          /* bytes sent/received */
                                          /* before secret key is */
                                          /* reset */
    MQLONG     FipsRequired;           /* Using FIPS-certified */
                                          /* algorithms */
    /* Ver:2 */
    MQLONG     EncryptionPolicySuiteB[4]; /* Use only Suite B */
    /* Ver:3 */
    MQLONG     CertificateValPolicy;    /* cryptographic algorithms */
                                          /* Certificate validation */
                                          /* policy */
    /* Ver:4 */
    MQCHAR64   CertificateLabel;       /* Certificate label */
    /* Ver:5 */
    MQPTR      KeyRepoPasswordPtr;     /* Address of key */
                                          /* repository password */
    MQLONG     KeyRepoPasswordOffset;  /* Offset of key repository */
                                          /* password */
    MQLONG     KeyRepoPasswordLength;  /* Length of key repository */
                                          /* password */
    /* Ver:6 */
};
```

### Déclaration COBOL pour MQSCO

```
** MQSCO structure
10 MQSCO.
** Structure identifier
15 MQSCO-STRUCID PIC X(4).
** Structure version number
15 MQSCO-VERSION PIC S9(9) BINARY.
** Location of TLS key repository
15 MQSCO-KEYREPOSITORY PIC X(256).
** Cryptographic hardware configuration string
15 MQSCO-CRYPTOHardware PIC X(256).
** Number of MQAIR records present
15 MQSCO-AUTHINFORECCOUNT PIC S9(9) BINARY.
** Offset of first MQAIR record from start of MQSCO structure
15 MQSCO-AUTHINFORECOFFSET PIC S9(9) BINARY.
** Address of first MQAIR record
15 MQSCO-AUTHINFORECPtr POINTER.
** Version 1 **
** Number of unencrypted bytes sent/received before secret key is
** reset
15 MQSCO-KEYRESETCOUNT PIC S9(9) BINARY.
** Using FIPS-certified algorithms
```

```

15 MQSCO-FIPSREQUIRED          PIC S9(9) BINARY.
** Version 2 **
**   Use only Suite B cryptographic algorithms
15 MQSCO-ENCRYPTIONPOLICYSUITEB PIC S9(9) BINARY OCCURS 4.
** Version 3 **
**   Certificate validation policy setting
15 MQSCO-CERTIFICATEVALPOLICY   PIC S9(9) BINARY.
** Version 4 **
**   SSL/TLS certificate label
15 MQSCO-CERTIFICATELABEL       PIC X(64).
** Version 5 **
**   Add padding to ensure that pointers start on correct
**   boundaries
15 FILLER                       PIC S9(9) BINARY VALUE 0.
**   Address of key repository password
15 MQSCO-KEYREPOPASSWORDPTR     POINTER.
**   Offset of key repository password
15 MQSCO-KEYREPOPASSWORDOFFSET  PIC S9(9) BINARY.
**   Length of key repository password
15 MQSCO-KEYREPOPASSWORDLENGTH  PIC S9(9) BINARY.
** Version 6 **

```

## Déclaration PL/I pour MQSCO

```

dcl
  1 MQSCO based,
    3 StrucId          char(4),          /* Structure identifier */
    3 Version         fixed bin(31),    /* Structure version number */
    3 KeyRepository   char(256),        /* Location of TLS key
                                        repository */
    3 CryptoHardware  char(256),        /* Cryptographic hardware
                                        configuration string */
    3 AuthInfoRecCount fixed bin(31),   /* Number of MQAIR records
                                        present */
    3 AuthInfoRecOffset fixed bin(31), /* Offset of first MQAIR record
                                        from start of MQSCO structure */
    3 AuthInfoRecPtr  pointer,          /* Address of first MQAIR record */
    3 KeyResetCount   fixed bin(31),   /* Key reset count */
/* Version 1 */
    3 FipsRequired    fixed bin(31),   /* FIPS required */
/* Version 2 */
    3 EncryptionPolicySuiteB (4) fixed bin(31), /* Suite B encryption policy */
/* Version 3 */
    3 CertificateValPolicy fixed bin(31), /* Certificate validation policy */
/* Version 4 */
    3 CertificateLabel char(64),        /* SSL/TLS certificate label */
/* Version 5 */
    3 KeyRepoPasswordPtr pointer,        /* Address of key repository
                                        password */
    3 KeyRepoPasswordOffset fixed bin(31), /* Offset of key repository
                                        password */
    3 KeyRepoPasswordLength fixed bin(31); /* Length of key repository
                                        password */
/* Version 6 */

```

## Déclaration Visual Basic pour MQSCO

```

Type MQSCO
  StrucId          As String*4  'Structure identifier'
  Version         As Long      'Structure version number'
  KeyRepository   As String*256 'Location of TLS key repository'
  CryptoHardware  As String*256 'Cryptographic hardware configuration'
  AuthInfoRecCount As Long      'Number of MQAIR records present'
  AuthInfoRecOffset As Long     'Offset of first MQAIR record from'
  AuthInfoRecPtr  As MQPTR     'start of MQSCO structure'
  AuthInfoRecPtr  As MQPTR     'Address of first MQAIR record'
  KeyResetCount   As Long      'Number of unencrypted bytes sent/received before secret key
  is reset'
  'Version 1'
  FipsRequired    As Long      'Mandatory FIPS CipherSpecs?'
  'Version 2'
End Type

```

## Référence associée

«MQCNO-Options de connexion», à la page 325

La structure MQCNO permet à l'application de spécifier des options relatives à la connexion au gestionnaire de files d'attente. La structure est un paramètre d'entrée-sortie sur l'appel MQCONN.

### **StrucId (MQCHAR4) pour MQSCO**

Il s'agit de l'identificateur de structure de la structure des options de configuration SSL/TLS. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQSCO\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUC\_MQSCO**

Identificateur de la structure des options de configuration SSL/TLS.

Pour le langage de programmation C, la constante MQSCO\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQSCO\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQSCO**

Il s'agit du numéro de version de la structure ; la valeur doit être:

#### **MQSCO\_VERSION\_1**

Structure des options de configuration TLS Version-1 .

#### **MQSCO\_VERSION\_2**

Structure des options de configuration TLS Version-2 .

#### **MQSCO\_VERSION\_3**

Structure des options de configuration TLS Version-3 .

#### **MQSCO\_VERSION\_4**

Structure des options de configuration TLS Version-4 .

#### **MQSCO\_VERSION\_5**

Structure des options de configuration TLS Version-5 .

#### **MQSCO\_VERSION\_6**

Structure des options de configuration TLS Version-6 .

La constante suivante indique le numéro de version de la version en cours:

#### **VERSION\_MQSCO\_CURRENT\_VERSION**

Version actuelle de la structure des options de configuration TLS.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQSCO\_VERSION\_1.

### **Multi KeyRepository (MQCHAR256) pour MQSCO**

Cette zone est pertinente uniquement pour les IBM MQ MQI clients s'exécutant sur des systèmes IBM i, AIX, Linux, and Windows . Il indique l'emplacement du fichier de la base de données de clés dans lequel les clés et les certificats sont stockés. Si le suffixe de fichier n'est pas spécifié, le suffixe .kdb est ajouté automatiquement.

Chaque fichier de base de données de clés peut être associé à un *fichier de mot de passe secret*. Le fichier de dissimulation contient les mots de passe codés qui sont utilisés pour autoriser l'accès par programmation à la base de données de clés. Le fichier de mot de passe secret doit résider dans le même répertoire et avoir le même radical de fichier que la base de données de clés, et doit se terminer par le suffixe .sth.

Par exemple, si le fichier de la base de données de clés est /xxx/yyy/key.kdb, le fichier de dissimulation de mot de passe doit être /xxx/yyy/key.sth, où xxx et yyy représentent les noms de répertoire.

Le mot de passe de la base de données de clés peut également être spécifié à l'aide de la zone *KeyRepoPasswordPtr* ou *KeyRepoPasswordOffset* de la structure MQSCO.

Si la valeur est inférieure à la longueur de la zone, terminez la valeur par un caractère NULL ou remplissez la zone avec des blancs. La valeur n'est pas vérifiée ; en cas d'erreur lors de l'accès au référentiel de clés, l'appel échoue avec le code anomalie MQRC\_KEY\_REPOSITORY\_ERROR.

Pour exécuter une connexion TLS à partir d'un IBM MQ MQI client, définissez *KeyRepository* sur un nom de fichier de base de données de clés valide.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par MQ\_SSL\_KEY\_REPOSITORY\_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.

### **CryptoHardware (MQCHAR256) pour MQSCO**

Cette zone fournit des détails de configuration pour le matériel cryptographique connecté au système client.

Définissez la zone sur une chaîne au format suivant, ou laissez-la vide ou null:

```
GSK_PKCS11=the PKCS #11 driver path and file name;the PKCS #11 token label;the PKCS #11 token password;symmetric cipher setting;
```

Pour utiliser du matériel cryptographique conforme à l'interface PKCS #11 , par exemple IBM 4960 ou IBM 4764, le chemin d'accès au pilote PKCS #11 , le libellé de jeton PKCS #11 et les chaînes de mot de passe de jeton PKCS #11 doivent être indiqués, chacune se terminant par un point-virgule.

Le chemin du pilote PKCS #11 est un chemin d'accès absolu à la bibliothèque partagée prenant en charge la carte PKCS #11 . Le nom de fichier du pilote PKCS #11 est le nom de la bibliothèque partagée. Voici un exemple de la valeur requise pour le chemin d'accès et le nom de fichier PKCS #11 :

```
/usr/lib/pkcs11/PKCS11_API.so
```

Le libellé de jeton PKCS #11 doit correspondre au libellé avec lequel vous avez configuré votre matériel.

Si aucune configuration de matériel de cryptographie n'est requise, définissez la zone sur une valeur vide ou null.

Si la valeur est inférieure à la longueur de la zone, terminez la valeur par un caractère NULL ou remplissez la zone avec des blancs. Si la valeur n'est pas valide ou entraîne un échec lors de l'utilisation de la configuration du matériel de cryptographie, l'appel échoue avec le code anomalie MQRC\_CRYPTO\_HARDWARE\_ERROR.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par MQ\_SSL\_CRYPTO\_HARDWARE\_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.

### **AuthInfoRecCount (MQLONG) pour MQSCO**

Il s'agit du nombre d'enregistrements d'informations d'authentification (MQAIR) traités par les zones *AuthInfoRecPtr* ou *AuthInfoRecOffset* . Pour plus d'informations, voir «MQAIR-Enregistrement des informations d'authentification», à la page 277. La valeur doit être supérieure ou égale à zéro. Si la valeur n'est pas valide, l'appel échoue avec le code anomalie MQRC\_AUTH\_INFO\_REC\_COUNT\_ERROR.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

### **AuthInfoRecOffset (MQLONG) pour MQSCO**

Décalage en octets du premier enregistrement d'informations d'authentification à partir du début de la structure MQSCO. Le décalage peut être positif ou négatif. La zone est ignorée si *AuthInfoRecCount* a la valeur zéro.

Vous pouvez utiliser *AuthInfoRecOffset* ou *AuthInfoRecPtr* pour spécifier les enregistrements MQAIR, mais pas les deux ; voir la description de la zone *AuthInfoRecPtr* pour plus de détails.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

### ***AuthInfoRecPtr (PMQAIR) pour MQSCO***

Il s'agit de l'adresse du premier enregistrement d'informations d'authentification. La zone est ignorée si *AuthInfoRecCount* a la valeur zéro.

Vous pouvez fournir le tableau d'enregistrements MQAIR de l'une des deux manières suivantes:

- A l'aide de la zone de pointeur *AuthInfoRecPtr*

Dans ce cas, l'application peut déclarer un tableau d'enregistrements MQAIR distinct de la structure MQSCO et définir *AuthInfoRecPtr* sur l'adresse du tableau.

Envisagez d'utiliser *AuthInfoRecPtr* pour les langages de programmation qui prennent en charge le type de données de pointeur d'une manière portable dans différents environnements (par exemple, le langage de programmation C).

- En utilisant la zone de décalage *AuthInfoRecOffset*

Dans ce cas, l'application doit déclarer une structure composée contenant un MQSCO suivi du tableau d'enregistrements MQAIR et définir *AuthInfoRecOffset* sur le décalage du premier enregistrement du tableau à partir du début de la structure MQSCO. Vérifiez que cette valeur est correcte et qu'elle peut être prise en compte dans un MQLONG (le langage de programmation le plus restrictif est COBOL, pour lequel la plage valide est comprise entre -999 999 999 et +999 999 999).

Envisagez d'utiliser *AuthInfoRecOffset* pour les langages de programmation qui ne prennent pas en charge le type de données de pointeur ou qui implémentent le type de données de pointeur d'une manière qui n'est pas portable dans des environnements différents (par exemple, le langage de programmation COBOL).

Quelle que soit la technique choisie, vous ne pouvez utiliser qu'une seule des options *AuthInfoRecPtr* et *AuthInfoRecOffset* ; l'appel échoue avec le code anomalie MQRC\_AUTH\_INFO\_REC\_ERROR si les deux sont différents de zéro.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire.

**Remarque :** Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

### ***KeyReset(MQLONG) pour MQSCO***

Représente le nombre total d'octets non chiffrés envoyés et reçus dans une conversation TLS avant la renégociation de la clé secrète.

Le nombre d'octets inclut les informations de contrôle envoyées par l'agent MCA.

Si vous spécifiez un nombre de réinitialisations de clé confidentielle TLS compris entre 1 octet et 32 Ko, les canaux TLS utiliseront un nombre de réinitialisations de clé confidentielle de 32 Ko. Cela permet d'éviter le coût de traitement d'un trop grand nombre de réinitialisations de clé qui se produiraient pour des valeurs de réinitialisation de clé secrète TLS de petite taille.

Il s'agit d'une zone d'entrée. La valeur est un nombre compris entre 0 et 999 999 999, avec une valeur par défaut de 0. Utilisez la valeur 0 pour indiquer que les clés secrètes ne sont jamais renégociées.

### ***FipsRequired (MQLONG) pour MQSCO***

IBM MQ peut être configuré avec du matériel de cryptographie de sorte que les modules de cryptographie utilisés soient ceux fournis par le produit matériel ; ils peuvent être certifiés FIPS à un niveau particulier en fonction du produit matériel de cryptographie utilisé. Utilisez cette zone pour indiquer que seuls les algorithmes certifiés FIPS sont utilisés si la cryptographie est fournie dans un logiciel fourni par IBM MQ.

**Remarque :** Sous AIX, Linux, and Windows, IBM MQ fournit la conformité à la norme FIPS 140-2 via le module cryptographique IBM Crypto for C (ICC) . Le certificat de ce module a été déplacé vers le statut Historique. Les clients doivent afficher le [certificat IBM Crypto for C \(ICC\)](#) et prendre connaissance des

conseils fournis par le NIST. Un module FIPS 140-3 de remplacement est actuellement en cours et son statut peut être affiché en le recherchant dans la [liste des modules NIST CMVP en cours de traitement](#).

Lorsque IBM MQ est installé, une implémentation de la cryptographie TLS est également installée, qui fournit des modules certifiés FIPS.

Les valeurs possibles sont les suivantes:

#### **MQSSL\_FIPS\_NO**

Il s'agit de la valeur par défaut. Lorsque cette valeur est définie:

- Tout CipherSpec pris en charge sur une plateforme particulière peut être utilisé.
- En cas d'exécution sans utilisation de matériel de cryptographie, les CipherSpecs s'exécutent à l'aide de la cryptographie certifiée FIPS 140-2 sur les plateformes IBM MQ .

Pour obtenir la liste des CipherSpecs certifiés FIPS, voir le tableau décrit dans [Activation des CipherSpecs](#).

#### **MQSSL\_FIPS\_YES**

Lorsque cette valeur est définie, sauf si vous utilisez du matériel de cryptographie pour effectuer la cryptographie, vous pouvez être certain que

- Seuls les algorithmes de cryptographie certifiés FIPS peuvent être utilisés dans le CipherSpec s'appliquant à cette connexion client.
- Les connexions de canal TLS entrantes et sortantes aboutissent uniquement si certaines spécifications de chiffrement sont utilisées.

Pour plus d'informations, voir [Activation des CipherSpecs CipherSpecs](#) .

**Remarque :** Dans la mesure du possible, si des CipherSpecs FIPS uniquement sont configurés, le client MQI rejette les connexions qui spécifient un CipherSpec non FIPS avec MQRC\_SSL\_INITIALIZATION\_ERROR. IBM MQ ne garantit pas le rejet de toutes ces connexions et il vous incombe de déterminer si votre configuration IBM MQ est conforme à la norme FIPS.

#### ***EncryptionPolicySuiteB (MQLONG) pour MQSCO***

Cette zone indique si la cryptographie compatible Suite B est utilisée et quel niveau de puissance est utilisé. La valeur peut être une ou plusieurs des valeurs suivantes:

- MQ\_SUITE\_B\_NONE

La cryptographie conforme à la suite B n'est pas utilisée.

- MQ\_SUITE\_B\_128\_BIT

La sécurité de la suite B 128 bits est utilisée.

- MQ\_SUITE\_B\_192\_BIT

La sécurité de la suite B 192 bits est utilisée.

**Remarque :** L'utilisation de MQ\_SUITE\_B\_NONE avec une autre valeur de cette zone n'est pas valide.

#### ***Règle CertificateVal(MQLONG) pour MQSCO***

Cette zone indique le type de règle de validation de certificat utilisé.

La zone peut être définie sur l'une des valeurs suivantes:

#### **MQ\_CERT\_VAL\_POLICY\_ANY**

Appliquez chacune des règles de validation de certificat prises en charge par la bibliothèque de sockets sécurisés. Acceptez la chaîne de certificats si l'une des politiques considère que la chaîne de certificats est valide.

#### **MQ\_CERT\_VAL\_POLICY\_RFC5280**

Appliquez uniquement la règle de validation de certificat RFC5280 . Ce paramètre fournit une validation plus stricte que le paramètre ANY, mais rejette certains certificats numériques plus anciens.



La valeur initiale de cette zone est MQ\_CERT\_VAL\_POLICY\_ANY

### ***CertificateLabel (MQCHAR64) pour MQSCO***

Cette zone fournit des détails sur le label de certificat utilisé.

IBM MQ initialise la valeur par défaut de la zone *CertificateLabel* sous la forme de blancs.

Cette valeur est interprétée lors de l'exécution comme la valeur par défaut et est compatible en amont.

Par exemple, si vous spécifiez une version de MQSCO antérieure à 5.0 ou que vous utilisez la valeur par défaut des blancs pour la zone *CertificateLabel*, la valeur par défaut préexistante de *ibmwebspheremquser\_id* est utilisée.

### ***KeyRepoPasswordPtr (MQPTR) pour MQSCO***

Il s'agit de l'adresse en octets de la phrase passe du référentiel de clés TLS.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire. Cette zone est ignorée si *Version* est inférieur à MQSCO\_VERSION\_6.

Cette zone est pertinente uniquement pour les IBM MQ MQI clients s'exécutant sur des systèmes IBM i, AIX, Linux, and Windows .

La phrase passe du référentiel de clés peut être spécifiée sous la forme d'une chaîne de texte en clair ou d'une phrase passe qui a été chiffrée à l'aide de l'utilitaire **runmqicred** .

Si vous fournissez une phrase passe chiffrée, indiquez la clé initiale qui a été utilisée pour chiffrer la phrase passe dans la structure MQCSP fournie par la même application client.

La phrase passe du référentiel de clés spécifiée à l'aide de cette zone remplace toute phrase passe du référentiel de clés spécifiée à l'aide de la variable d'environnement *MQKEYRPWD* ou de la propriété *SSLKeyRepositoryPassword* dans la section SSL du fichier de configuration du client.

Vous pouvez utiliser *KeyRepoPasswordOffset* ou *KeyRepoPasswordPtr* pour spécifier la phrase passe du référentiel de clés, mais pas les deux.

#### **Tâches associées**

[Fourniture d'une clé initiale pour un client IBM MQ MQI sous AIX, Linux et Windows](#)

[Protection des mots de passe dans les fichiers de configuration du composant IBM MQ](#)

#### **Référence associée**

[runmqicred \(protection des mots de passe du client IBM MQ\)](#)

«InitialKeyPtr (MQPTR) pour MQCSP», à la page 353

[Adresse de la clé initiale du système de protection par mot de passe.](#)

### ***KeyRepoPasswordOffset (MQLONG) pour MQSCO***

Décalage en octets de la phrase passe du référentiel de clés TLS à partir du début de la structure MQSCO. Le décalage peut être positif ou négatif.

Vous pouvez utiliser *KeyRepoPasswordOffset* ou *KeyRepoPasswordPtr* pour spécifier la phrase passe du référentiel de clés, mais pas les deux. Pour plus d'informations, voir la description de la zone *KeyRepoPasswordPtr* .

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQSCO\_VERSION\_6.

### ***KeyRepoPasswordLength (MQLONG) pour MQSCO***

Longueur de la phrase passe du référentiel de clés TLS.

Sous IBM i, la longueur maximale de la phrase passe du référentiel de clés est de 128 caractères. Si la phrase passe du référentiel de clés est supérieure à la longueur maximale autorisée, la connexion échoue avec MQRC\_KEY\_REPOSITORY\_ERROR.





Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQSCO\_VERSION\_6.

## MQSD - Descripteur d'abonnement

La structure MQSD est utilisée pour spécifier les détails de l'abonnement effectué. La structure est un paramètre d'entrée-sortie dans l'appel MQSUB. Pour plus d'informations, voir les [remarques sur l'utilisation de MQSUB](#).

### Disponibilité

La structure MQSD est disponible sur les plateformes suivantes:

-  AIX
-  IBM i
-  Linux
-  Windows
-  z/OS

et pour IBM MQ MQI clients connecté à ces systèmes.

### Version

La version actuelle de MQSD est MQSD\_VERSION\_1.

### Jeu de caractères et codage

Les données dans MQSD doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et le codage du gestionnaire de files d'attente local indiqué par MQENC\_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

### Abonnements gérés

Si une application n'a pas besoin d'utiliser une file d'attente particulière comme destination pour les publications qui correspondent à son abonnement, elle peut utiliser la fonction d'abonnement géré. Si une application choisit d'utiliser un abonnement géré, le gestionnaire de files d'attente informe l'abonné de la destination où les messages publiés sont envoyés, en fournissant un descripteur d'objet comme sortie de l'appel MQSUB. Pour plus d'informations, voir [Hobj \(MQHOBJ\)-input/output](#).

Lorsque l'abonnement est supprimé, le gestionnaire de files d'attente s'engage également à nettoyer les messages qui n'ont pas été extraits de la destination gérée, dans les situations suivantes:

- Lorsque l'abonnement est supprimé-à l'aide de MQCLOSE avec MQCO\_REMOVE\_SUB-et que le Hobj géré est fermé.
- Par des moyens implicites lorsque la connexion est perdue à une application à l'aide d'un abonnement non durable (MQSO\_NON\_DURABLE)
- Par expiration lorsqu'un abonnement est supprimé parce qu'il est arrivé à expiration et que le Hobj géré est fermé.

Vous devez utiliser des abonnements gérés avec des abonnements non durables afin que ce nettoyage puisse se produire et que les messages des abonnements non durables fermés n'occupent pas d'espace dans votre gestionnaire de files d'attente. Les abonnements durables peuvent également utiliser des destinations gérées.

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID_STRUCD_MQS	'SD↵↵'
<u>Version</u> (numéro de version de la structure)	MQSD_VERSION_1	1
<u>Options</u> (options)	MQSO_NON_DURABLE	0
<u>ObjectName</u> (nom de l'objet)	Aucun	Chaîne nulle ou blancs
<u>IDAlternateUser</u> (autre ID utilisateur)	Aucun	Chaîne nulle ou blancs
<u>IDAlternateSecurity</u> (autre ID de sécurité)	MQSID_AUCUN	Valeurs NULL
<u>SubExpiry</u> (expiration de l'abonnement)	MQEI_UNLIMITED	-1
<u>ObjectString</u> (chaîne d'objet)	Aucun	Noms et valeurs définis pour MQCHARV
<u>SubName</u> (nom de l'abonnement)	Aucun	Noms et valeurs définis pour MQCHARV
<u>Données SubUser</u> (données utilisateur d'abonnement)	Aucun	Noms et valeurs définis pour MQCHARV
<u>IDSubCorrel</u> (ID corrélation d'abonnement)	MQCI_NONE	Valeurs NULL
<u>PubPriority</u> (priorité de publication)	MQPRI_PRIORITY_AS_Q_DEF	-3
<u>PubAccountingToken</u> (jeton de comptabilité de publication)	MQACT_AUCUN	Valeurs NULL
<u>PubAppIdentityData</u> (données d'identité de l'application de publication)	Aucun	Chaîne nulle ou blancs
<u>SelectionString</u> (chaîne fournissant des critères de sélection)	Aucun	Noms et valeurs définis pour MQCHARV
<u>SubLevel</u> (niveau d'abonnement)	Aucun	1
<u>ResObjectString</u> (nom d'objet long)	Aucun	Noms et valeurs définis pour MQCHARV

### Remarques :

1. Le symbole ↵ représente un caractère blanc unique.
2. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.
3. Dans le langage de programmation C, la variable macroMQSD\_DEFAULT contient les valeurs répertoriées dans la table. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQSD MySD = {MQSD_DEFAULT};
```

## Déclarations de langue

### Déclaration C pour MQSD

```
typedef struct tagMQSD MQSD;
struct tagMQSD {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;          /* Structure version number */
    MQLONG     Options;          /* Options associated with subscribing */
    MQCHAR48   ObjectName;       /* Object name */
    MQCHAR12   AlternateUserId;  /* Alternate user identifier */
    MQBYTE40   AlternateSecurityId; /* Alternate security identifier */
    MQLONG     SubExpiry;        /* Expiry of Subscription */
    MQCHARV   ObjectString;      /* Object Long name */
    MQCHARV   SubName;           /* Subscription name */
    MQCHARV   SubUserData;       /* Subscription User data */
    MQBYTE24   SubCorrelId;      /* Correlation Id related to this subscription */
    MQLONG     PubPriority;       /* Priority set in publications */
    MQBYTE32   PubAccountingToken; /* Accounting Token set in publications */
    MQCHAR32   PubApplIdentityData; /* Appl Identity Data set in publications */
    MQCHARV   SelectionString;   /* Message selector structure */
    MQLONG     SubLevel;         /* Subscription level */
    MQCHARV   ResObjectString;   /* Resolved Long object name*/
    /* Ver:1 */
};
```

### Déclaration COBOL pour MQSD

```
** Address of variable length string
20 MQSD-OBJECTSTRING-VSPTR          POINTER.
** Offset of variable length string
20 MQSD-OBJECTSTRING-VSOFFSET      PIC S9(9) BINARY.
** size of buffer
20 MQSD-OBJECTSTRING-VSBUFSIZE     PIC S9(9) BINARY.
** Length of variable length string
20 MQSD-OBJECTSTRING-VSLENGTH      PIC S9(9) BINARY.
** CCSID of variable length string
20 MQSD-OBJECTSTRING-VSCCSID      PIC S9(9) BINARY.
** Subscription name
15 MQSD-SUBNAME.
** Address of variable length string
20 MQSD-SUBNAME-VSPTR              POINTER.
** Offset of variable length string
20 MQSD-SUBNAME-VSOFFSET          PIC S9(9) BINARY.
** size of buffer
20 MQSD-SUBNAME-VSBUFSIZE         PIC S9(9) BINARY.
** Length of variable length string
20 MQSD-SUBNAME-VSLENGTH          PIC S9(9) BINARY.
** CCSID of variable length string
20 MQSD-SUBNAME-VSCCSID          PIC S9(9) BINARY.
** Subscription User data
15 MQSD-SUBUSERDATA.
** Address of variable length string
20 MQSD-SUBUSERDATA-VSPTR          POINTER.
** Offset of variable length string
20 MQSD-SUBUSERDATA-VSOFFSET      PIC S9(9) BINARY.
** size of buffer
20 MQSD-SUBUSERDATA-VSBUFSIZE     PIC S9(9) BINARY.
** Length of variable length string
20 MQSD-SUBUSERDATA-VSLENGTH      PIC S9(9) BINARY.
** CCSID of variable length string
20 MQSD-SUBUSERDATA-VSCCSID      PIC S9(9) BINARY.
** Correlation Id related to this subscription
15 MQSD-SUBCORRELID               PIC X(24).
** Priority set in publications
15 MQSD-PUBPRIORITY               PIC S9(9) BINARY.
** Accounting Token set in publications
15 MQSD-PUBACCOUNTINGTOKEN        PIC X(32).
** Appl Identity Data set in publications
15 MQSD-PUBAPPLIDENTITYDATA       PIC X(32).
** Message Selector
15 MQSD-SELECTIONSTRING.
** Address of variable length string
20 MQSD-SELECTIONSTRING-VSPTR      POINTER.
** Offset of variable length string
20 MQSD-SELECTIONSTRING-VSOFFSET   PIC S9(9) BINARY.
** size of buffer
```

```

20 MQSD-SELECTIONSTRING-VSBUFSIZE      PIC S9(9) BINARY.
** Length of variable length string
20 MQSD-SELECTIONSTRING-VSLENGTH      PIC S9(9) BINARY.
** CCSID of variable length string
20 MQSD-SELECTIONSTRING-VSCCSID      PIC S9(9) BINARY.
** Selection criteria
20 MQSD-SELECTIONSTRING-SUBLEVEL      PIC S9(9) BINARY.
** Long object name
20 MQSD-SELECTIONSTRING-RESOBJSTRING  PIC S9(9) BINARY.

```

## Déclaration PL/I pour MQSD

```

dcl
1 MQSD based,
3 StructId      char(4), /* Structure identifier */
3 Version       fixed bin(31), /* Structure version number */
3 Options       fixed bin(31), /* Options associated with subscribing */
3 ObjectName    char(48), /* Object name */
3 AlternateUserId char(12), /* Alternate user identifier */
3 AlternateSecurityId char(40), /* Alternate security identifier */
3 SubExpiry     fixed bin(31), /* Expiry of Subscription */
3 ObjectString, /* Object Long name */
5 VSPtr        pointer, /* Address of variable length string */
5 VSOffset     fixed bin(31), /* Offset of variable length string */
5 VSBufSize    fixed bin(31), /* size of buffer */
5 VSLength     fixed bin(31), /* Length of variable length string */
5 VSCCSID      fixed bin(31); /* CCSID of variable length string */
3 SubName, /* Subscription name */
5 VSPtr        pointer, /* Address of variable length string */
5 VSOffset     fixed bin(31), /* Offset of variable length string */
5 VSBufSize    fixed bin(31), /* size of buffer */
5 VSLength     fixed bin(31), /* Length of variable length string */
5 VSCCSID      fixed bin(31); /* CCSID of variable length string */
3 SubUserData, /* Subscription User data */
5 VSPtr        pointer, /* Address of variable length string */
5 VSOffset     fixed bin(31), /* Offset of variable length string */
5 VSBufSize    fixed bin(31), /* size of buffer */
5 VSLength     fixed bin(31), /* Length of variable length string */
5 VSCCSID      fixed bin(31), /* CCSID of variable length string */
3 SubCorrelId  char(24), /* Correlation Id related to this subscription */
3 PubPriority   fixed bin(31), /* Priority set in publications */
3 PubAccountingToken char(32), /* Accounting Token set in publications */
3 PubApplIdentityData char(32), /* Appl Identity Data set in publications */
3 SelectionString, /* Message Selection */
5 VSPtr        pointer, /* Address of variable length string */
5 VSOffset     fixed bin(31), /* Offset of variable length string */
5 VSBufSize    fixed bin(31), /* size of buffer */
5 VSLength     fixed bin(31), /* Length of variable length string */
5 VSCCSID      fixed bin(31), /* CCSID of variable length string */
3 SubLevel     fixed bin(31), /* Subscription level */
3 ResObjectString, /* Resolved Long object name */
5 VSPtr        pointer, /* Address of variable length string */
5 VSOffset     fixed bin(31), /* Offset of variable length string */
5 VSBufSize    fixed bin(31), /* size of buffer */
5 VSLength     fixed bin(31), /* Length of variable length string */
5 VSCCSID      fixed bin(31); /* CCSID of variable length string */

```

## Déclaration High Level Assembler pour MQSD

```

MQSD          DSECT
MQSD_STRUCID  DS CL4 Structure identifier
MQSD_VERSION  DS F Structure version number
MQSD-OPTIONS  DS F Options associated with subscribing
MQSD-OBJECTNAME DS CL48 Object name
MQSD_ALTERNATEUSERID DS CL12 Alternate user identifier
MQSD_ALTERNATESECURITYID DS CL40 Alternate security identifier
MQSD_SUBEXPIRY DS F Expiry of Subscription
MQSD-OBJECTSTRING DS 0F Object Long name
MQSD-OBJECTSTRING_VSPTR DS F Address of variable length string
MQSD-OBJECTSTRING_VSOFFSET DS F Offset of variable length string
MQSD-OBJECTSTRING_VSBUFSIZE DS F size of buffer
MQSD-OBJECTSTRING_VSLENGTH DS F Length of variable length string
MQSD-OBJECTSTRING_VSCCSID DS F CCSID of variable length string
MQSD-OBJECTSTRING_LENGTH EQU *-MQSD-OBJECTSTRING
ORG MQSD-OBJECTSTRING
MQSD-OBJECTSTRING_AREA DS CL(MQSD-OBJECTSTRING_LENGTH)
*
MQSD-SUBNAME DS 0F Subscription name
MQSD-SUBNAME_VSPTR DS F Address of variable length string

```

```

MQSD_SUBNAME_VSOFFSET    DS  F   Offset of variable length string
MQSD_SUBNAME_VSBUFSIZE  DS  F   size of buffer
MQSD_SUBNAME_VSLENGTH   DS  F   Length of variable length string
MQSD_SUBNAME_VSCCSID    DS  F   CCSID of variable length string
MQSD_SUBNAME_LENGTH     EQU  *-MQSD_SUBNAME
ORG  MQSD_SUBNAME
MQSD_SUBNAME_AREA       DS  CL(MQSD_SUBNAME_LENGTH)
*
MQSD_SUBUSERDATA        DS  0F   Subscription User data
MQSD_SUBUSERDATA_VSPTR  DS  F   Address of variable length string
MQSD_SUBUSERDATA_VSOFFSET DS  F   Offset of variable length string
MQSD_SUBUSERDATA_VSBUFSIZE DS  F   size of buffer
MQSD_SUBUSERDATA_VSLENGTH DS  F   Length of variable length string
MQSD_SUBUSERDATA_VSCCSID DS  F   CCSID of variable length string
MQSD_SUBUSERDATA_LENGTH EQU  *-MQSD_SUBUSERDATA
ORG  MQSD_SUBUSERDATA
MQSD_SUBUSERDATA_AREA   DS  CL(MQSD_SUBUSERDATA_LENGTH)
*
MQSD_SUBCORRELID        DS  CL24 Correlation Id related to this subscription
MQSD_PUBPRIORITY        DS  F   Priority set in publications
MQSD_PUBACCOUNTINGTOKEN DS  CL32 Accounting Token set in publications
MQSD_PUBAPPLIDENTITYDATA DS  CL32 Appl Identity Data set in publications
*
MQSD_SELECTIONSTRING    DS  F   Message Selector
MQSD_SELECTIONSTRING_VSPTR DS  F   Address of variable length string
MQSD_SELECTIONSTRING_VSOFFSET DS  F   Offset of variable length string
MQSD_SELECTIONSTRING_VSBUFSIZE DS  F   size of buffer
MQSD_SELECTIONSTRING_VSLENGTH DS  F   Length of variable length string
MQSD_SELECTIONSTRING_VSCCSID DS  F   CCSID of variable length string
MQSD_SELECTIONSTRING_LENGTH EQU  *-MQSD_SELECTIONSTRING
ORG  MQSD_SELECTIONSTRING
MQSD_SELECTIONSTRING_AREA DS  CL(MQSD_SELECTIONSTRING_LENGTH)
*
MQSD-SUBLEVEL           DS  F   Subscription level
*
MQSD_RESOBJECTSTRING     DS  F   Resolved Long object name
MQSD_RESOBJECTSTRING_VSPTR DS  F   Address of variable length string
MQSD_RESOBJECTSTRING_VSOFFSET DS  F   Offset of variable length string
MQSD_RESOBJECTSTRING_VSBUFSIZE DS  F   size of buffer
MQSD_RESOBJECTSTRING_VSLENGTH DS  F   Length of variable length string
MQSD_RESOBJECTSTRING_VSCCSID DS  F   CCSID of variable length string
MQSD_RESOBJECTSTRING_LENGTH EQU  *-MQSD_RESOBJECTSTRING
ORG  MQSD_RESOBJECTSTRING
MQSD_RESOBJECTSTRING_AREA DS  CL(MQSD_RESOBJECTSTRING_LENGTH)
*
MQSD_LENGTH             EQU  *-MQSD
ORG  MQSD
MQSD_AREA               DS  CL(MQSD_LENGTH)

```

### **StrucId (MQCHAR4) pour MQSD**

Il s'agit de l'identificateur de structure de la structure de descripteur d'abonnement. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQSD\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUCD\_MQS**

Identificateur de la structure de descripteur d'abonnement.

Pour le langage de programmation C, la constante MQSD\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQSD\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQSD**

Il s'agit du numéro de version de la structure ; la valeur doit être:

#### **MQSD\_VERSION\_1**

Structure de descripteur d'abonnement Version-1 .

La constante suivante indique le numéro de version de la version en cours:

#### **MQSD\_CURRENT\_VERSION**

Version actuelle de la structure du descripteur d'abonnement.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQSD\_VERSION\_1.

## Options (MQLONG) pour MQSD

Fournit des options permettant de contrôler l'action de l'appel MQSUB.

Vous devez spécifier au moins l'une des options suivantes:

- MQSO\_ALTER
- MQSO\_RESUME
- MQSO\_CREER

Pour spécifier plusieurs options, ajoutez les valeurs ensemble (n'ajoutez pas la même constante plusieurs fois) ou combinez les valeurs à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations bit).

Les combinaisons qui ne sont pas valides sont indiquées dans cette rubrique ; les autres combinaisons sont valides.

**Options d'accès ou de création:** les options d'accès et de création déterminent si un abonnement est créé ou si un abonnement existant est renvoyé ou modifié. Vous devez spécifier au moins l'une de ces options.

Combinaison d'options	Remarques
MQSO_CREER	Crée un abonnement s'il n'en existe pas. Cette combinaison échoue si l'abonnement existe.
MQSO_RESUME	Reprend un abonnement existant. Cette combinaison échoue si aucun abonnement n'existe.
MQSO_CREATE + MQSO_RESUME	Crée un abonnement s'il n'en existe pas et en reprend un correspondant, s'il existe. Cette combinaison est utile lorsqu'elle est utilisée dans une application qui est exécutée plusieurs fois.
MQSO_ALTER (voir la remarque)	Reprend un abonnement existant, en modifiant toutes les zones pour qu'elles correspondent à celles spécifiées dans le MQSD. Cette combinaison échoue si aucun abonnement n'existe.
MQSO_CREATE + MQSO_ALTER (voir la remarque)	Crée un abonnement s'il n'en existe pas et reprend un abonnement correspondant, s'il existe, en modifiant les zones pour qu'elles correspondent à celles spécifiées dans le MQSD. Cette combinaison est utile lorsqu'elle est utilisée dans une application qui souhaite s'assurer que son abonnement est dans un certain état avant de continuer.

### Remarque :

Les options spécifiant MQSO\_ALTER peuvent également spécifier MQSO\_RESUME, mais cette combinaison n'a pas d'effet supplémentaire sur la spécification de MQSO\_ALTER uniquement. MQSO\_ALTER implique MQSO\_RESUME, car l'appel de MQSUB pour modifier un abonnement implique que l'abonnement sera également repris. Le contraire n'est pas vrai, cependant: la reprise d'un abonnement n'implique pas qu'il doit être modifié.

### MQSO\_CREER

Créez un nouvel abonnement pour la rubrique spécifiée. Si un abonnement utilisant le même *SubName* existe, l'appel échoue avec MQRC\_SUB\_ALREADY\_EXISTS. Cet échec peut être évité en combinant l'option MQSO\_CREATE avec MQSO\_RESUME. Le *SubName* n'est pas toujours nécessaire. Pour plus de détails, voir la description de cette zone.

La combinaison de MQSO\_CREATE avec MQSO\_RESUME renvoie un descripteur à un abonnement préexistant pour le *SubName* spécifié s'il en existe un ; s'il n'existe pas d'abonnement existant, un nouveau descripteur est créé à l'aide de toutes les zones fournies dans le MQSD.

MQSO\_CREATE peut également être combiné avec MQSO\_ALTER pour obtenir un effet similaire.

## **MQSO\_RESUME**

Renvoie un descripteur à un abonnement préexistant qui correspond à celui spécifié par *SubName*. Aucune modification n'est apportée aux attributs d'abonnement correspondants et ils sont renvoyés en sortie dans la structure MQSD. Seules les zones MQSD suivantes sont utilisées: StrucId, Version, Options, AlternateUserId et AlternateSecurityId et SubName.

L'appel échoue avec le code anomalie MQRC\_NO\_SUBSCRIPTION si un abonnement ne correspond pas au nom d'abonnement complet. Cet échec peut être évité en combinant l'option MQSO\_CREATE avec MQSO\_RESUME.

L'ID utilisateur de l'abonnement est l'ID utilisateur qui a créé l'abonnement, ou s'il a été modifié ultérieurement par un autre ID utilisateur, il s'agit de l'ID utilisateur de la modification ayant abouti la plus récente. Si un ID AlternateUser est utilisé et que l'utilisation d'ID utilisateur de remplacement est autorisée pour cet utilisateur, l'ID utilisateur de remplacement est enregistré en tant qu'ID utilisateur qui a créé l'abonnement à la place de l'ID utilisateur sous lequel l'abonnement a été effectué.

S'il existe un abonnement correspondant qui a été créé sans l'option MQSO\_ANY\_USERID et que l'ID utilisateur de l'abonnement est différent de celui de l'application demandant un descripteur à l'abonnement, l'appel échoue avec le code anomalie MQRC\_IDENTITY\_MISMATCH.

Si un abonnement correspondant existe et est en cours d'utilisation, l'appel échoue avec MQRC\_SUBSCRIPTION\_IN\_USE.

Si l'abonnement nommé dans *SubName* n'est pas un abonnement valide à reprendre ou à modifier à partir d'une application, l'appel échoue avec MQRC\_INVALID\_SUBSCRIPTION.

MQSO\_RESUME étant implicite dans MQSO\_ALTER, vous n'avez pas besoin de le combiner avec cette option. Toutefois, la combinaison des deux options ne génère pas d'erreur.

## **MQSO\_ALTER**

Renvoie un descripteur à un abonnement préexistant dont le nom d'abonnement complet correspond à celui spécifié par le nom dans *SubName*. Tous les attributs de l'abonnement qui sont différents de ceux spécifiés dans le MQSD sont modifiés dans l'abonnement sauf si la modification est interdite pour cet attribut. Les détails sont indiqués dans la description de chaque attribut et sont résumés dans le tableau suivant. Si vous tentez de modifier un attribut qui ne peut pas être modifié ou de modifier un abonnement qui a défini l'option MQSO\_IMMUTABLE, l'appel échoue avec le code anomalie indiqué dans le tableau suivant.

L'appel échoue avec le code anomalie MQRC\_NO\_SUBSCRIPTION si un abonnement correspondant au nom d'abonnement complet n'existe pas. Vous pouvez éviter cet échec en combinant l'option MQSO\_CREATE avec MQSO\_ALTER.

La combinaison de MQSO\_CREATE et de MQSO\_ALTER renvoie un descripteur à un abonnement préexistant pour le *SubName* spécifié s'il en existe un ; s'il n'existe pas d'abonnement existant, un nouveau descripteur est créé à l'aide de toutes les zones fournies dans le MQSD.

L'ID utilisateur de l'abonnement est l'ID utilisateur qui a créé l'abonnement, ou s'il est modifié ultérieurement par un autre ID utilisateur, il s'agit de l'ID utilisateur de la modification la plus récente ayant abouti. Si un ID AlternateUser est utilisé et que l'utilisation d'ID utilisateur de remplacement est autorisée pour cet utilisateur, l'ID utilisateur de remplacement est enregistré en tant qu'ID utilisateur qui a créé l'abonnement à la place de l'ID utilisateur sous lequel l'abonnement a été effectué.

S'il existe un abonnement correspondant qui a été créé sans l'option MQSO\_ANY\_USERID et que l'ID utilisateur de l'abonnement est différent de celui de l'application demandant un descripteur à l'abonnement, l'appel échoue avec le code anomalie MQRC\_IDENTITY\_MISMATCH.

Si un abonnement correspondant existe et est en cours d'utilisation, l'appel échoue avec MQRC\_SUBSCRIPTION\_IN\_USE.



Si l'abonnement nommé dans SubName n'est pas un abonnement valide à reprendre ou à modifier à partir d'une application, l'appel échoue avec MQR\_INVALID\_SUBSCRIPTION.

Le tableau suivant montre la capacité de MQSO ALTER à modifier les valeurs d'attribut dans MQSD et MQSUB.

Tableau 527. Attributs dans MQSD et MQSUB pouvant être modifiés			
Descripteur de type de données ou appel de fonction	Nom de zone	Cet attribut peut-il être modifié à l'aide de MQSO ALTER	Code raison
Disque MQSD	Options de durabilité	Non	MQRC_DURATIONITY_NOT_ALTERABLE
Disque MQSD	Options de destination	Oui	Aucun
Disque MQSD	Options d'enregistrement	Oui (voir la remarque «1», à la page 601)	MQRC_GROUPING_NOT_ALTERABLE si vous tentez de modifier MQSO_GROUP_SUB
Disque MQSD	Options de publication	Oui (voir la remarque «2», à la page 601)	Aucun
Disque MQSD	Options de caractère générique	Non	MQRC_TOPIC_NOT_ALTERABLE
Disque MQSD	Autres options	Non (voir la remarque «3», à la page 601)	Aucun
Disque MQSD	ObjectName	Non	MQRC_TOPIC_NOT_ALTERABLE
Disque MQSD	AlternateUserid	Non (voir la remarque «4», à la page 601)	Aucun
Disque MQSD	AlternateSecurityId	Non (voir la remarque «4», à la page 601)	Aucun
Disque MQSD	SubExpiry	Oui	Aucun
Disque MQSD	ObjectString	Non	MQRC_TOPIC_NOT_ALTERABLE
Disque MQSD	SubName	Non (voir la remarque «5», à la page 601)	Aucun
Disque MQSD	SubUserData	Oui	Aucun
Disque MQSD	SubCorrelId	Oui (voir la remarque «6», à la page 601)	MQRC_GROUPING_NOT_ALTERABLE dans un abonnement groupé
Disque MQSD	PubPriority	Oui	Aucun
Disque MQSD	Jeton PubAccounting	Oui	Aucun
Disque MQSD	PubApplIdentityData	Oui	Aucun
Disque MQSD	SubLevel	Non	MQRC_SUBLEVEL_NOT_ALTERABLE
MQSUB	HOBJ	Oui (voir la remarque «6», à la page 601)	MQRC_GROUPING_NOT_ALTERABLE dans un abonnement groupé

#### Remarques :

1. MQSO\_GROUP\_SUB ne peut pas être modifié.
2. MQSO\_NEW\_PUBLICATIONS\_ONLY ne peut pas être modifié car il ne fait pas partie de l'abonnement
3. Ces options ne font pas partie de l'abonnement
4. Cet attribut ne fait pas partie de l'abonnement
5. Cet attribut correspond à l'identité de l'abonnement en cours de modification
6. Modifiable sauf lorsqu'il fait partie d'un sous-groupe (MQSO\_GROUP\_SUB)

**Options de durabilité:** Les options suivantes contrôlent la durabilité de l'abonnement. Vous ne pouvez spécifier qu'une seule de ces options. Si vous modifiez un abonnement existant à l'aide de l'option MQSO ALTER, vous ne pouvez pas modifier la durabilité de l'abonnement. En cas de retour d'un appel MQSUB à l'aide de MQSO\_RESUME, l'option de durabilité appropriée est définie.

#### MQSO DURABLE

Demandez que l'abonnement à cette rubrique soit conservé jusqu'à ce qu'elle soit explicitement supprimée à l'aide de MQCLOSE avec l'option MQCO\_REMOVE\_SUB. Si cet abonnement n'est pas

explicitement supprimé, il sera conservé même après la fermeture de cette connexion d'application au gestionnaire de files d'attente.

Si un abonnement durable est demandé à une rubrique définie comme n'autorisant pas les abonnements durables, l'appel échoue avec MQRC\_DURABILITY\_NOT\_ALLOWED.

### **MQSO\_NON\_DURABLE**

Demandez la suppression de l'abonnement à cette rubrique lorsque la connexion des applications au gestionnaire de files d'attente est fermée, si elle n'est pas déjà explicitement supprimée.

MQSO\_NON\_DURABLE est l'opposé de l'option MQSO\_DURABLE et est défini pour aider la documentation du programme. Il s'agit de la valeur par défaut si aucune n'est spécifiée.

**Options de destination:** L'option suivante contrôle la destination à laquelle les publications d'une rubrique à laquelle vous avez souscrit sont envoyées. Si vous modifiez un abonnement existant à l'aide de l'option MQSO\_ALTER, la destination utilisée pour les publications de l'abonnement peut être modifiée. En cas de retour d'un appel MQSUB à l'aide de MQSO\_RESUME, cette option est définie si nécessaire.

### **MQSO\_GERE**

Demande que la destination à laquelle les publications sont envoyées soit gérée par le gestionnaire de files d'attente.

L'identificateur d'objet renvoyé dans *Hobj* représente une file d'attente gérée par le gestionnaire de files d'attente et doit être utilisé avec les appels MQGET, MQCB, MQINQ ou MQCLOSE ultérieurs.

Un descripteur d'objet renvoyé par un appel MQSUB précédent ne peut pas être fourni dans le paramètre **Hobj** lorsque MQSO\_MANAGED n'est pas spécifié.

### **MQSO\_NO\_MULTICAST**

Demandez que la destination à laquelle les publications sont envoyées ne soit pas une adresse de groupe de multidiffusion. Cette option est valide uniquement lorsqu'elle est associée à l'option MQSO\_MANAGED. Lorsqu'un descripteur d'une file d'attente est fourni dans le paramètre **Hobj**, la multidiffusion ne peut pas être utilisée pour cet abonnement et l'option n'est pas valide.

Si la rubrique est définie pour n'autoriser que les abonnements multidiffusion, à l'aide du paramètre MCAST (ONLY), l'appel échoue avec le code anomalie MQRC\_MULTICAST\_REQUIRED.

**Option de portée:** L'option suivante contrôle la portée de l'abonnement effectué. Si vous modifiez un abonnement existant à l'aide de l'option MQSO\_ALTER, cette option de portée d'abonnement ne peut pas être modifiée. Lors du retour d'un appel MQSUB à l'aide de MQSO-RESUME, l'option de portée appropriée est définie.

### **MQSO\_SCOPE\_QMGR**

Cet abonnement est effectué uniquement sur le gestionnaire de files d'attente local. Aucun abonnement de proxy n'est distribué aux autres gestionnaires de files d'attente du réseau. Seules les publications publiées sur ce gestionnaire de files d'attente sont envoyées à cet abonné. Cette option remplace tout comportement défini à l'aide de l'attribut de rubrique SUBSCOPE.

**Remarque :** S'il n'est pas défini, la portée de l'abonnement est déterminée par l'attribut de rubrique SUBSCOPE.

**Options d'enregistrement:** Les options suivantes contrôlent les détails de l'enregistrement effectué auprès du gestionnaire de files d'attente pour cet abonnement. Si vous modifiez un abonnement existant à l'aide de l'option MQSO\_ALTER, ces options d'enregistrement peuvent être modifiées. En cas de retour d'un appel MQSUB à l'aide de MQSO\_RESUME, les options d'enregistrement appropriées sont définies.

### **SOUS-GROUPE\_MQSO**

Cet abonnement doit être regroupé avec d'autres abonnements du même SubLevel à l'aide de la même file d'attente et en spécifiant le même ID de corrélation de sorte que toute publication destinée à des rubriques qui entraînerait la fourniture de plusieurs messages de publication au groupe d'abonnements, en raison du chevauchement d'un ensemble de chaînes de rubrique utilisées, entraîne la distribution d'un seul message à la file d'attente. Si cette option n'est pas utilisée, chaque abonnement unique (identifié par SubName) qui correspond est fourni avec une copie de

la publication, ce qui peut signifier que plusieurs copies de la publication peuvent être placées dans la file d'attente partagée par un certain nombre d'abonnements.

Seul l'abonnement le plus important du groupe est fourni avec une copie de la publication. L'abonnement le plus significatif est basé sur le nom de rubrique complet jusqu'au point où un caractère générique est trouvé. Si une combinaison de schémas génériques est utilisée dans le groupe, seule la position du caractère générique est importante. Il est conseillé de ne pas combiner différents schémas de caractères génériques dans un groupe d'abonnements partageant la même file d'attente.

Lors de la création d'un abonnement groupé, il doit toujours avoir un SubNameunique, mais s'il correspond au nom de rubrique complet d'un abonnement existant dans le groupe, l'appel échoue avec MQRD\_DUPLICATE\_GROUP\_SUB.

Si l'abonnement le plus significatif dans le groupe spécifie également MQSO\_NOT\_OWN\_PUBS et qu'il s'agit d'une publication provenant de la même application, aucune publication n'est distribuée dans la file d'attente.

Lors de la modification d'un abonnement effectué avec cette option, les zones qui impliquent le regroupement, Hobj sur l'appel MQSUB (représentant le nom de la file d'attente et du gestionnaire de files d'attente) et l'ID SubCorrelne peuvent pas être modifiées. La tentative de modification entraîne l'échec de l'appel avec MQRD\_GROUPING\_NOT\_ALTERABLE.

Cette option doit être combinée avec MQSO\_SET\_CORREL\_ID avec un ID SubCorrelqui n'est pas défini sur MQCI\_NONE et ne peut pas être combinée avec MQSO\_MANAGED.

### **ID utilisateur MQSO\_ANY\_USERID**

Lorsque MQSO\_ANY\_USERID est spécifié, l'identité de l'abonné n'est pas limitée à un seul ID utilisateur. Cela permet à tout utilisateur de modifier ou de reprendre l'abonnement lorsqu'il dispose des droits appropriés. Un seul utilisateur peut disposer de l'abonnement à la fois. Une tentative de reprise de l'utilisation d'un abonnement actuellement utilisé par une autre application entraîne l'échec de l'appel avec MQRD\_SUBSCRIPTION\_IN\_USE.

Pour ajouter cette option à un abonnement existant, l'appel MQSUB (à l'aide de MQSO\_ALTER) doit provenir du même ID utilisateur que l'abonnement d'origine lui-même.

Si un appel MQSUB fait référence à un abonnement existant avec MQSO\_ANY\_USERID défini et que l'ID utilisateur est différent de l'abonnement d'origine, l'appel aboutit uniquement si le nouvel ID utilisateur est autorisé à s'abonner à la rubrique. Une fois l'opération terminée, les futures publications destinées à cet abonné sont placées dans la file d'attente des abonnés avec le nouvel ID utilisateur défini dans le message de publication.

N'indiquez pas à la fois MQSO\_ANY\_USERID et MQSO\_FIXED\_USERID. Si aucun des deux n'est spécifié, la valeur par défaut est MQSO\_FIXED\_USERID.

### **ID\_UTILISATEUR\_FIXE\_MQSO\_FIXE**

Lorsque MQSO\_FIXED\_USERID est spécifié, l'abonnement peut être modifié ou repris uniquement par le dernier ID utilisateur pour modifier l'abonnement. Si l'abonnement n'a pas été modifié, il s'agit de l'ID utilisateur qui a créé l'abonnement.

Si une instruction MQSUB fait référence à un abonnement existant avec MQSO\_ANY\_USERID défini et modifie l'abonnement à l'aide de MQSO\_ALTER pour utiliser l'option MQSO\_FIXED\_USERID, l'ID utilisateur de l'abonnement est désormais corrigé au niveau de ce nouvel ID utilisateur. L'appel aboutit uniquement si le nouvel ID utilisateur est autorisé à s'abonner à la rubrique.

Si un ID utilisateur autre que celui enregistré comme propriétaire d'un abonnement tente de reprendre ou de modifier un abonnement MQSO\_FIXED\_USERID, l'appel échoue avec MQRD\_IDENTITY\_MISMATCH. L'ID utilisateur propriétaire d'un abonnement peut être affiché à l'aide de la commande DISPLAY SBSTATUS.

N'indiquez pas à la fois MQSO\_ANY\_USERID et MQSO\_FIXED\_USERID. Si aucun des deux n'est spécifié, la valeur par défaut est MQSO\_FIXED\_USERID.

**Options de publication:** Les options suivantes contrôlent la manière dont les publications sont envoyées à cet abonné. Si vous modifiez un abonnement existant à l'aide de l'option MQSO ALTER, vous pouvez modifier ces options de publication.

### **MQSO\_NON\_PUBS**

Indique au courtier que l'application ne souhaite pas voir ses propres publications. Les publications sont considérées comme provenant de la même application si les descripteurs de connexion sont identiques. En cas de retour d'un appel MQSUB à l'aide de MQSO\_RESUME, cette option est définie si nécessaire.

### **MQSO\_NEW\_PUBLICATIONS\_ONLY**

Aucune publication actuellement conservée ne doit être envoyée, lorsque cet abonnement est créé, uniquement les nouvelles publications. Cette option s'applique uniquement lorsque MQSO\_CREATE est spécifié. Les modifications apportées ultérieurement à un abonnement ne modifient pas le flux des publications et, par conséquent, les publications conservées sur une rubrique ont déjà été envoyées à l'abonné en tant que nouvelles publications.

Si cette option est spécifiée sans MQSO\_CREATE, l'appel échoue avec MQRC\_OPTIONS\_ERROR. En cas de retour d'un appel MQSUB à l'aide de MQSO\_RESUME, cette option n'est pas définie même si l'abonnement a été créé à l'aide de cette option.

Si cette option n'est pas utilisée, les messages précédemment conservés sont envoyés à la file d'attente de destination fournie. Si cette action échoue en raison d'une erreur, MQRC\_RETAINED\_MSG\_Q\_ERROR ou MQRC\_RETAINED\_NOT\_LIVRÉES, la création de l'abonnement échoue.

### **MQSO\_PUBLICATIONS\_ON\_REQUEST**

La définition de cette option indique que l'abonné demandera des informations spécifiquement lorsque cela est nécessaire. Le gestionnaire de files d'attente n'envoie pas de messages non sollicités à l'abonné. La publication conservée (ou éventuellement plusieurs publications si un caractère générique est spécifié dans la rubrique) est envoyée à l'abonné chaque fois qu'un appel MQSUBRQ est effectué à l'aide du descripteur Hsub d'un appel MQSUB précédent. Aucune publication n'est envoyée suite à l'appel MQSUB utilisant cette option. En cas de retour d'un appel MQSUB à l'aide de MQSO\_RESUME, cette option est définie si nécessaire.

Cette option n'est pas valide avec un SubLevel supérieur à 1.

**Options de lecture anticipée:** Les options suivantes contrôlent si les messages non persistants sont envoyés à une application avant l'application qui les demande.

### **MQSO\_READ\_AHEAD\_AS\_Q\_DEF**

Si l'appel MQSUB utilise un descripteur géré, l'attribut de lecture anticipée par défaut de la file d'attente modèle associée à la rubrique à laquelle l'utilisateur est abonné détermine si les messages sont envoyés à l'application avant que l'application ne les demande.

Il s'agit de la valeur par défaut.

### **MQSO\_NO\_READ\_AHEAD**

Si l'appel MQSUB utilise un descripteur géré, les messages ne sont pas envoyés à l'application avant que celle-ci ne les demande.

### **MQSO\_READ\_AHEAD**

Si l'appel MQSUB utilise un descripteur géré, des messages peuvent être envoyés à l'application avant que celle-ci ne les demande.

### **Remarque :**

Les remarques suivantes s'appliquent aux options de lecture anticipée:

1. Une seule de ces options peut être spécifiée. Si MQSO\_READ\_AHEAD et MQSO\_NO\_READ\_AHEAD sont spécifiées, le code anomalie MQRC\_OPTIONS\_ERROR est renvoyé. Ces options ne sont applicables que si MQSO\_MANAGED est spécifié.
2. Elles ne s'appliquent pas à MQSUB lorsqu'une file d'attente qui a été précédemment ouverte est transmise. Il se peut que la lecture anticipée ne soit pas activée lorsqu'elle est demandée. Les options

MQGET utilisées lors du premier appel MQGET peuvent empêcher l'activation de la lecture anticipée. En outre, la lecture anticipée est désactivée lorsque le client se connecte à un gestionnaire de files d'attente où la lecture anticipée n'est pas prise en charge. Si l'application n'est pas exécutée en tant que client IBM MQ, ces options sont ignorées.

**Options de caractères génériques:** Les options suivantes contrôlent la façon dont les caractères génériques sont interprétés dans la chaîne fournie dans la zone ObjectString du MQSD. Vous ne pouvez spécifier qu'une seule de ces options. Si vous modifiez un abonnement existant à l'aide de l'option MQSO\_ALTER, ces options génériques ne peuvent pas être modifiées. En cas de retour d'un appel MQSUB à l'aide de MQSO\_RESUME, l'option de caractère générique appropriée est définie.

### MQSO\_WILDCARD\_CHAR

Les caractères génériques ne s'appliquent qu'aux caractères de la chaîne de rubrique.

Le comportement défini par MQSO\_WILDCARD\_CHAR est illustré dans le tableau suivant.

Caractère spécial	Comportement
la barre oblique (/)	Pas de signification, juste un autre caractère
Astérisque (*)	Caractère générique, zéro ou plusieurs caractères
Point d'interrogation (?)	Caractère générique, 1 caractère
le symbole de pourcentage (%)	Caractère d'échappement permettant aux caractères (*), (?) ou (%) d'être utilisés dans une chaîne et de ne pas être interprétés comme un caractère spécial, par exemple (% *), (%?) ou (%%).

Par exemple, la publication sur la rubrique suivante:

```
/level0/level1/level2/level3/level4
```

correspond aux abonnés à l'aide des rubriques suivantes:

```
*
/*
/ level0/level1/level2/level3/*
/ level0/level1/*/level3/level4
/ level0/level1/le?e12/level3/level4
```

**Remarque :** Cette utilisation de caractères génériques fournit exactement la signification fournie dans IBM MQ V6 et WebSphere MB V6 lors de l'utilisation de messages MQRFH1 formatés pour la publication / l'abonnement. Il est recommandé de ne pas l'utiliser pour les applications nouvellement écrites et de l'utiliser uniquement pour les applications qui étaient précédemment exécutées sur cette version et qui n'ont pas été modifiées pour utiliser le comportement de caractère générique par défaut, comme décrit dans MQSO\_WILDCARD\_TOPIC.

### MQSO\_WILDCARD\_TOPIC

Les caractères génériques n'agissent que sur les éléments de rubrique dans la chaîne de rubrique. Il s'agit du comportement par défaut si aucun n'est sélectionné.

Le comportement requis par MQSO\_WILDCARD\_TOPIC est présenté dans le tableau suivant:

Caractère spécial	Comportement
(/)	Séparateur de niveau de rubrique
Signe dièse (#)	Caractère générique: plusieurs niveaux de rubrique

Tableau 529. Interprétation des caractères génériques (suite)

Caractère spécial	Comportement
Signe plus (+)	Caractère générique: niveau de rubrique unique

**Remarques :**

Les caractères (+) et (#) ne sont pas traités comme des caractères génériques s'ils sont mélangés à d'autres caractères (y compris eux-mêmes) dans un niveau de rubrique. Dans la chaîne suivante, les caractères (#) et (+) sont traités comme des caractères ordinaires.

```
level0/level1/#+/level3/level#
```

Par exemple, la publication sur la rubrique suivante:

```
/level0/level1/level2/level3/level4
```

correspond aux abonnés à l'aide des rubriques suivantes:

```
#
/#
/ level0/level1/level2/level3/#
/ level0/level1+/level3/level4
```

**Autres options:** Les options suivantes contrôlent la manière dont l'appel d'API est émis et non l'abonnement. En cas de retour d'un appel MQSUB à l'aide de MQSO\_RESUME, ces options restent inchangées. Pour plus d'informations, voir «ID AlternateUser(MQCHAR12) pour MQSD», à la page 608.

**MQSO\_ALTERNATE\_USER\_AUTHORITY**

La zone ID AlternateUser contient un identificateur utilisateur à utiliser pour valider cet appel MQSUB. L'appel peut aboutir uniquement si cet ID AlternateUser est autorisé à ouvrir l'objet avec les options d'accès spécifiées, que l'ID utilisateur sous lequel l'application s'exécute soit ou non autorisé à le faire.

**ID\_CORREL\_SET\_MQSO**

L'abonnement doit utiliser l'identificateur de corrélation fourni dans la zone *SubCorrelId*. Si cette option n'est pas spécifiée, un identificateur de corrélation est automatiquement créé par le gestionnaire de files d'attente au moment de l'abonnement et renvoyé à l'application dans la zone *SubCorrelId*. Pour plus d'informations, voir «ID SubCorrel(MQBYTE24) pour MQSD», à la page 610.

Cette option ne peut pas être combinée avec MQSO\_MANAGED.

**MQSO\_SET\_IDENTITY\_CONTEXT**

L'abonnement doit utiliser le jeton de comptabilité et les données d'identité d'application fournies dans les zones *PubAccountingToken* et *PubAppIdentityData*.

Si cette option est spécifiée, la même vérification d'autorisation est effectuée comme si la file d'attente de destination était accessible à l'aide d'un appel MQOPEN avec MQOO\_SET\_IDENTITY\_CONTEXT, sauf dans le cas où l'option MQSO\_MANAGED est également utilisée, auquel cas aucune vérification d'autorisation n'est effectuée sur la file d'attente de destination.

Si cette option n'est pas spécifiée, les informations de contexte par défaut sont associées aux publications envoyées à cet abonné comme suit:

<i>Tableau 530. Informations de contexte par défaut pour les publications envoyées à cet abonné</i>	
<b>Zone dans MQMD</b>	<b>Valeur utilisée</b>
<i>UserIdentifier</i>	ID utilisateur associé à l'abonnement au moment où l'abonnement a été effectué.
<i>AccountingToken</i>	Déterminé à partir de l'environnement si possible ; défini sur MQACT_NONE si ce n'est pas le cas.
<i>ApplIdentityData</i>	Mettre à blanc

Cette option est valide uniquement avec MQSO\_CREATE et MQSO\_ALTER. Si elle est utilisée avec MQSO\_RESUME, les zones *PubAccountingToken* et *PubApplIdentityData* sont ignorées, de sorte que cette option n'a aucun effet.

Si un abonnement est modifié sans utiliser cette option alors que les informations de contexte d'identité fournies par l'abonnement étaient précédemment fournies, des informations de contexte par défaut sont générées pour l'abonnement modifié.

Si un abonnement permettant à différents ID utilisateur de l'utiliser avec l'option MQSO\_ANY\_USERID, est repris par un autre ID utilisateur, le contexte d'identité par défaut est généré pour le nouvel ID utilisateur propriétaire de l'abonnement et toutes les publications suivantes sont distribuées contenant le nouveau contexte d'identité.

### **MQSO\_FAIL\_IF QUIESCING**

L'appel MQSUB échoue si le gestionnaire de files d'attente est à l'état de mise au repos. Sous z/OS, pour une application CICS ou IMS, cette option force également l'échec de l'appel MQSUB si la connexion est à l'état de mise au repos.

### **ObjectName (MQCHAR48) pour MQSD**

Il s'agit du nom de l'objet de rubrique tel qu'il est défini sur le gestionnaire de files d'attente local.

Le nom peut contenir les caractères suivants:

- Caractères alphabétiques en majuscules (A à Z)
- Caractères alphabétiques minuscules (a à z)
- Chiffres (0 à 9)
- Point (.), barre oblique (/), trait de soulignement (\_), pourcentage (%)

Le nom ne doit pas contenir d'espaces de début ou d'espaces imbriqués, mais peut contenir des espaces de fin. Utilisez un caractère null pour indiquer la fin des données significatives dans le nom ; la valeur null et les caractères qui la suivent sont traités comme des blancs. Les restrictions suivantes s'appliquent dans les environnements indiqués:

- Sur les systèmes qui utilisent EBCDIC Katakana, les caractères minuscules ne peuvent pas être utilisés.
- Sous z/OS :
  - Evitez les noms qui commencent ou se terminent par un trait de soulignement ; ils ne peuvent pas être traités par les panneaux d'opérations et de contrôle.
  - Le caractère de pourcentage a une signification spéciale pour RACF. Si RACF est utilisé comme gestionnaire de sécurité externe, les noms ne doivent pas contenir le pourcentage. Si tel est le cas, ces noms ne sont pas inclus dans les contrôles de sécurité lorsque des profils génériques RACF sont utilisés.
- Sous IBM i, les noms contenant des caractères minuscules, des barres obliques ou des pourcentages doivent être placés entre guillemets lorsqu'ils sont spécifiés dans les commandes. Ces guillemets ne doivent pas être indiqués pour les noms qui apparaissent en tant que zones dans les structures ou en tant que paramètres dans les appels.

*ObjectName* est utilisé pour former le nom complet de la rubrique.

Le nom complet de la rubrique peut être généré à partir de deux zones différentes: *ObjectName* et *ObjectString*. Pour plus de détails sur l'utilisation de ces deux zones, voir [Combinaison de chaînes de rubrique](#).

Si l'objet identifié par la zone *ObjectName* est introuvable, l'appel échoue avec le code anomalie MQRC\_UNKNOWN\_OBJECT\_NAME même si une chaîne est spécifiée dans *ObjectString*.

En cas de retour d'un appel MQSUB à l'aide de l'option MQSO\_RESUME, cette zone est inchangée.

La longueur de cette zone est indiquée par MQ\_TOPIC\_NAME\_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

Si vous modifiez un abonnement existant à l'aide de l'option MQSO\_ALTER, le nom de l'objet de rubrique auquel vous êtes abonné ne peut pas être modifié. Cette zone et la zone *ObjectString* peuvent être omises. S'ils sont fournis, ils doivent être résolus avec le même nom de rubrique complet. Si ce n'est pas le cas, l'appel échoue avec MQRC\_TOPIC\_NOT\_ALTERABLE.

### **ID AlternateUser(MQCHAR12) pour MQSD**

Si vous spécifiez MQSO\_ALTERNATE\_USER\_AUTHORITY, cette zone contient un autre identificateur utilisateur qui est utilisé pour vérifier l'autorisation pour l'abonnement et la sortie dans la file d'attente de destination (spécifiée dans le paramètre **Hobj** de l'appel MQSUB), à la place de l'identificateur utilisateur sous lequel l'application est en cours d'exécution.

Si l'opération aboutit, l'ID utilisateur indiqué dans cette zone est enregistré en tant qu'ID utilisateur propriétaire de l'abonnement à la place de l'ID utilisateur sous lequel l'application est en cours d'exécution.

Si MQSO\_ALTERNATE\_USER\_AUTHORITY est spécifié et que cette zone est entièrement vide jusqu'au premier caractère null ou jusqu'à la fin de la zone, l'abonnement peut aboutir uniquement si aucune autorisation utilisateur n'est nécessaire pour s'abonner à cette rubrique avec les options spécifiées ou la file d'attente de destination pour la sortie.

Si MQSO\_ALTERNATE\_USER\_AUTHORITY n'est pas spécifié, cette zone est ignorée.

Les différences suivantes existent dans les environnements indiqués:

- Sous z/OS, seuls les 8 premiers caractères de l'ID AlternateUser sont utilisés pour vérifier l'autorisation de l'abonnement. Toutefois, l'ID utilisateur en cours doit être autorisé à indiquer cet ID utilisateur de remplacement particulier ; les 12 caractères de l'ID utilisateur de remplacement sont utilisés pour cette vérification. L'ID utilisateur ne doit contenir que des caractères autorisés par le gestionnaire de sécurité externe.

En cas de retour d'un appel MQSUB à l'aide de MQSO\_RESUME, cette zone n'est pas modifiée.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par MQ\_USER\_ID\_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 12 caractères blancs dans les autres langages de programmation.

### **ID AlternateSecurity(MQBYTE40) pour MQSD**

Il s'agit d'un identificateur de sécurité qui est transmis avec l'ID AlternateUser au service d'autorisation pour permettre l'exécution des vérifications d'autorisation appropriées.

L'ID AlternateSecurity est utilisé uniquement si MQSO\_ALTERNATE\_USER\_AUTHORITY est spécifié et que la zone d'ID AlternateUser n'est pas entièrement vide jusqu'au premier caractère null ou jusqu'à la fin de la zone.

En cas de retour d'un appel MQSUB à l'aide de MQSO\_RESUME, cette zone n'est pas modifiée.



Pour plus d'informations, voir la description de «[ID AlternateSecurity\(MQBYTE40\) pour MQOD](#)», à la page 513 dans le type de données MQOD.

### ***SubExpiry (MQLONG) pour MQSD***

Il s'agit du temps exprimé en dixièmes de seconde après lequel l'abonnement expire. Aucune autre publication ne correspondra à cet abonnement une fois cet intervalle écoulé. Dès qu'un abonnement arrive à expiration, les publications ne sont plus envoyées à la file d'attente. Toutefois, les publications qui existent déjà ne sont en aucun cas affectées. *SubExpiry* n'a aucun effet sur l'expiration de la publication.

La valeur spéciale suivante est reconnue:

#### **MQEI\_UNLIMITED**

L'abonnement a un délai d'expiration illimité.

Si vous modifiez un abonnement existant à l'aide de l'option MQSO\_ALTER, l'expiration de l'abonnement peut être modifiée.

En cas de retour d'un appel MQSUB à l'aide de l'option MQSO\_RESUME, cette zone est définie sur l'expiration d'origine de l'abonnement et non sur le délai d'expiration restant.

### ***ObjectString (MQCHARV) pour MQSD***

Il s'agit du nom d'objet long à utiliser.

*ObjectString* est utilisé pour former le nom de rubrique complet.

Le nom complet de la rubrique peut être généré à partir de deux zones différentes: *ObjectName* et *ObjectString*. Pour plus de détails sur l'utilisation de ces deux zones, voir [Combinaison de chaînes de rubrique](#).

La longueur maximale de *ObjectString* est 10240.

Si *ObjectString* n'est pas spécifié correctement, conformément à la description de l'utilisation de la structure MQCHARV, ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie MQRC\_OBJECT\_STRING\_ERROR.

Il s'agit d'une zone d'entrée. Les valeurs initiales des zones de cette structure sont identiques à celles de la structure MQCHARV.

S'il existe des caractères génériques dans *ObjectString*, l'interprétation de ces caractères génériques peut être contrôlée à l'aide des options de caractères génériques spécifiées dans la zone Options du MQSD.

En cas de retour d'un appel MQSUB à l'aide de l'option MQSO\_RESUME, cette zone est inchangée. Le nom de rubrique complet utilisé est renvoyé dans la zone *ResObjectString* si une mémoire tampon est fournie.

Si vous modifiez un abonnement existant à l'aide de l'option MQSO\_ALTER, le nom long de l'objet de rubrique auquel vous êtes abonné ne peut pas être modifié. Cette zone et la zone *ObjectName* peuvent être omises. S'ils sont fournis, ils doivent être résolus avec le même nom de rubrique complet ou l'appel échoue avec MQRC\_TOPIC\_NOT\_ALTERABLE.

### ***SubName (MQCHARV) pour MQSD***

Indique le nom de l'abonnement. Cette zone est requise uniquement si *Options* spécifie l'option MQSO\_DURABLE, mais si elle est fournie, elle sera également utilisée par le gestionnaire de files d'attente pour MQSO\_NON\_DURABLE.

S'il est spécifié, *SubName* doit être unique dans le gestionnaire de files d'attente, car il s'agit de la méthode utilisée pour identifier l'abonnement.

La longueur maximale de *SubName* est 10240.

Cette zone a deux objectifs. Pour un abonnement MQSO\_DURABLE, vous utilisez cette zone pour identifier un abonnement afin de pouvoir le reprendre après sa création si vous avez fermé l'identificateur de l'abonnement (à l'aide de l'option MQCO\_KEEP\_SUB) ou si vous avez été déconnecté du gestionnaire de files d'attente. Cette opération est effectuée à l'aide de l'appel MQSUB avec l'option MQSO\_RESUME. Il est également affiché dans la vue d'administration des abonnements dans la zone SUBID de DISPLAY SBSTATUS.

Si *SubName* est spécifié de manière incorrecte, selon la description de la façon d'utiliser la structure MQCHARV, elle est laissée de côté lorsqu'elle est requise (c'est-à-dire *SubName.VSLength* est égal à zéro), ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie MQRC\_SUB\_NAME\_ERROR.

Il s'agit d'une zone d'entrée. Les valeurs initiales des zones de cette structure sont identiques à celles de la structure MQCHARV.

Si vous modifiez un abonnement existant à l'aide de l'option MQSO\_ALTER, le nom de l'abonnement ne peut pas être modifié car il s'agit de la zone d'identification utilisée pour rechercher l'abonnement référencé. Il n'est pas modifié dans la sortie d'un appel MQSUB avec l'option MQSO\_RESUME.

### **Données SubUser(MQCHARV) pour MQSD**

Indique les données utilisateur de l'abonnement. Les données fournies sur l'abonnement dans cette zone seront incluses en tant que propriété de message de données MQSubUserde chaque publication envoyée à cet abonnement.

La longueur maximale de *SubUserData* est 10240.

Si *SubUserData* est spécifié de manière incorrecte, conformément à la description de l'utilisation de la structure MQCHARV, ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie MQRC\_SUB\_USER\_DATA\_ERROR.

Il s'agit d'une zone d'entrée. Les valeurs initiales des zones de cette structure sont identiques à celles de la structure MQCHARV.

Si vous modifiez un abonnement existant à l'aide de l'option MQSO\_ALTER, les données utilisateur de l'abonnement peuvent être modifiées.

Cette zone de longueur variable est renvoyée en sortie d'un appel MQSUB à l'aide de l'option MQSO\_RESUME, si une mémoire tampon est fournie et qu'il existe une longueur de mémoire tampon positive dans *VSBuflen*. Si aucune mémoire tampon n'est fournie sur l'appel, seule la longueur de la date utilisateur de l'abonnement est renvoyée dans la zone *VSLength* de MQCHARV. Si la mémoire tampon fournie est inférieure à l'espace requis pour renvoyer la zone, seuls *VSBuflen* octets sont renvoyés dans la mémoire tampon fournie.

### **ID SubCorrel(MQBYTE24) pour MQSD**

Cette zone contient un identificateur de corrélation commun à toutes les publications correspondant à cet abonnement.



**Attention :** un identificateur de corrélation ne peut être transmis qu'entre des gestionnaires de files d'attente dans un cluster de publication / abonnement et non une hiérarchie.

Toutes les publications envoyées pour correspondre à cet abonnement contiennent cet identificateur de corrélation dans le descripteur de message. Si plusieurs abonnements obtiennent leurs publications de la même file d'attente, l'utilisation de MQGET par identificateur de corrélation permet d'obtenir uniquement les publications d'un abonnement spécifique. Cet identificateur de corrélation peut être généré par le gestionnaire de files d'attente ou par l'utilisateur.

Si l'option MQSO\_SET\_CORREL\_ID n'est pas spécifiée, l'identificateur de corrélation est généré par le gestionnaire de files d'attente et cette zone est une zone de sortie contenant l'identificateur de corrélation qui sera défini dans chaque message publié pour cet abonnement. L'identificateur de corrélation généré se compose d'un identificateur de produit de 4 octets (AMQX ou CSQM en ASCII ou EBCDIC) suivi d'une implémentation spécifique du produit d'une chaîne unique.

Si l'option MQSO\_SET\_CORREL\_ID est spécifiée, l'identificateur de corrélation est généré par l'utilisateur et cette zone est une zone d'entrée contenant l'identificateur de corrélation à définir dans chaque publication pour cet abonnement. Dans ce cas, si la zone contient MQCI\_NONE, l'identificateur de corrélation défini dans chaque message publié pour cet abonnement est l'identificateur de corrélation créé par l'insertion d'origine du message.

Si l'option MQSO\_GROUP\_SUB est spécifiée et que l'identificateur de corrélation spécifié est identique à un abonnement groupé existant utilisant la même file d'attente et une chaîne de rubrique se chevauchant, seul l'abonnement le plus significatif du groupe est fourni avec une copie de la publication.

La longueur de cette zone est indiquée par MQ\_CORREL\_ID\_LENGTH. La valeur initiale de cette zone est MQCI\_NONE.

Si vous modifiez un abonnement existant à l'aide de l'option MQSO\_ALTER et que cette zone est une zone d'entrée, l'identificateur de corrélation d'abonnement peut être modifié, sauf s'il s'agit d'un abonnement groupé, c'est-à-dire qu'il a été créé à l'aide de l'option MQSO\_GROUP\_SUB, auquel cas l'identificateur de corrélation d'abonnement ne peut pas être modifié.

En cas de retour d'un appel MQSUB à l'aide de MQSO\_RESUME, cette zone est définie sur l'identificateur de corrélation en cours de l'abonnement.

### **PubPriority (MQLONG) pour MQSD**

Il s'agit de la valeur qui sera dans la zone *Priority* du descripteur de message (MQMD) de tous les messages de publication correspondant à cet abonnement. Pour plus d'informations sur la zone *Priority* dans le MQMD, voir «[Priorité \(MQLONG\) pour MQMD](#)», à la page 470.

La valeur doit être supérieure ou égale à zéro ; zéro est la priorité la plus basse. Les valeurs spéciales suivantes peuvent également être utilisées:

#### **MQPRI\_PRIORITY\_AS\_Q\_DEF**

Lorsqu'une file d'attente d'abonnement est fournie dans la zone *Hobj* de l'appel MQSUB et qu'elle n'est pas un descripteur géré, la priorité du message est extraite de l'attribut **DefPriority** de cette file d'attente. Si la file d'attente est une file d'attente de cluster ou qu'il existe plusieurs définitions dans le chemin de résolution de nom de file d'attente, la priorité est déterminée lorsque le message de publication est inséré dans la file d'attente, comme décrit pour «[Priorité \(MQLONG\) pour MQMD](#)», à la page 470.

Si l'appel MQSUB utilise un descripteur géré, la priorité du message est extraite de l'attribut **DefPriority** de la file d'attente modèle associée à la rubrique à laquelle il est abonné.

#### **MQPRI\_PRIORITY\_AS\_PUBLISHED**

La priorité du message est la priorité de la publication d'origine. Il s'agit de la valeur initiale de la zone.

Si vous modifiez un abonnement existant à l'aide de l'option MQSO\_ALTER, le *Priority* des messages de publication ultérieurs peut être modifié.

En cas de retour d'un appel MQSUB à l'aide de MQSO\_RESUME, cette zone est définie sur la priorité en cours utilisée pour l'abonnement.

### **Jeton PubAccounting(MQBYTE32) pour MQSD**

Il s'agit de la valeur qui sera dans la zone *AccountingToken* du descripteur de message (MQMD) de tous les messages de publication correspondant à cet abonnement. *AccountingToken* fait partie du contexte d'identité du message. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#). Pour plus d'informations sur la zone *AccountingToken* dans le MQMD, voir «[AccountingToken \(MQBYTE32\) pour MQMD](#)», à la page 478

Vous pouvez utiliser la valeur spéciale suivante pour la zone *PubAccountingToken* :

## **MQACT\_AUCUN**

Aucun jeton de comptabilité n'est spécifié.

La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQACT\_NONE\_ARRAY est également définie ; elle a la même valeur que MQACT\_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Si l'option MQSO\_SET\_IDENTITY\_CONTEXT n'est pas spécifiée, le jeton de comptabilité est généré par le gestionnaire de files d'attente en tant qu'informations de contexte par défaut et cette zone est une zone de sortie qui contient le *AccountingToken* qui sera défini dans chaque message publié pour cet abonnement.

Si l'option MQSO\_SET\_IDENTITY\_CONTEXT est spécifiée, le jeton de comptabilité est généré par l'utilisateur et cette zone est une zone d'entrée qui contient le *AccountingToken* à définir dans chaque publication pour cet abonnement.

La longueur de cette zone est indiquée par MQ\_ACCOUNTING\_TOKEN\_LENGTH. La valeur initiale de cette zone est MQACT\_NONE.

Si vous modifiez un abonnement existant à l'aide de l'option MQSO\_ALTER, la valeur de *AccountingToken* dans les messages de publication ultérieurs peut être modifiée.

En cas de retour d'un appel MQSUB à l'aide de MQSO\_RESUME, cette zone est définie sur le *AccountingToken* en cours utilisé pour l'abonnement.

## ***PubApplIdentityData (MQCHAR32) pour MQSD***

Il s'agit de la valeur qui se trouve dans la zone *ApplIdentityData* du descripteur de message (MQMD) de tous les messages de publication correspondant à cet abonnement. *ApplIdentityData* fait partie du contexte d'identité du message. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#). Pour plus d'informations sur la zone *ApplIdentityData* dans le MQMD, voir «[Données ApplIdentity\(MQCHAR32\) pour MQMD](#)», à la page 480

Si l'option MQSO\_SET\_IDENTITY\_CONTEXT n'est pas spécifiée, le *ApplIdentityData* défini dans chaque message publié pour cet abonnement est vide, en tant qu'informations de contexte par défaut.

Si l'option MQSO\_SET\_IDENTITY\_CONTEXT est spécifiée, *PubApplIdentityData* est généré par l'utilisateur et cette zone est une zone d'entrée qui contient le *ApplIdentityData* à définir dans chaque publication pour cet abonnement.

La longueur de cette zone est indiquée par MQ\_APPL\_IDENTITY\_DATA\_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et 32 caractères blancs dans les autres langages de programmation.

Si vous modifiez un abonnement existant à l'aide de l'option MQSO\_ALTER, le *ApplIdentityData* des messages de publication ultérieurs peut être modifié.

En cas de retour d'un appel MQSUB à l'aide de MQSO\_RESUME, cette zone est définie sur le *ApplIdentityData* en cours utilisé pour l'abonnement.

## ***SelectionString (MQCHARV) pour MQSD***

Il s'agit de la chaîne utilisée pour fournir les critères de sélection utilisés lors de l'abonnement aux messages d'une rubrique.

Cette zone de longueur variable est renvoyée en sortie d'un appel MQSUB à l'aide de l'option MQSO\_RESUME, si une mémoire tampon est fournie, ainsi qu'une longueur de mémoire tampon positive dans VSBufSize. Si aucune mémoire tampon n'est fournie sur l'appel, seule la longueur de la chaîne de sélection est renvoyée dans la zone VSLength de MQCHARV. Si la mémoire tampon fournie est inférieure à l'espace requis pour renvoyer la zone, seuls les octets VSBufSize sont renvoyés dans la mémoire tampon fournie.

Si *SelectionString* est spécifié de manière incorrecte, conformément à la description du mode d'utilisation de la structure «MQCHARV-Chaîne de longueur variable», à la page 303 , ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie MQRC\_SELECTION\_STRING\_ERROR.

L'utilisation de *SelectionString* est décrite dans [Sélecteurs](#).

### **SubLevel (MQLONG) pour MQSD**

Niveau associé à l'abonnement. Les publications ne sont distribuées à cet abonnement que si elles se trouvent dans l'ensemble des abonnements dont la valeur *SubLevel* la plus élevée est inférieure ou égale à la valeur *PubLevel* utilisée au moment de la publication. Toutefois, si une publication a été conservée, elle n'est plus disponible pour les abonnés de niveau supérieur car elle est republiée sur le site *PubLevel* 1.

La valeur doit être comprise entre zéro et 9. Zéro est la valeur la plus faible.

La valeur initiale de cette zone est 1.

Pour plus d'informations, voir [Intercepting publications](#).

Si vous modifiez un abonnement existant à l'aide de l'option MQSO\_ALTER, le *SubLevel* ne peut pas être modifié.

La combinaison d'un *SubLevel* avec une valeur supérieure à 1 avec l'option MQSO\_PUBLICATIONS\_ON\_REQUEST n'est pas autorisée.

En cas de retour d'un appel MQSUB à l'aide de MQSO\_RESUME, cette zone est définie sur le niveau en cours utilisé pour l'abonnement.

### **Chaîne ResObject(MQCHARV) pour MQSD**

Il s'agit du nom d'objet long une fois que le gestionnaire de files d'attente a résolu le nom fourni dans *ObjectName*.

Si le nom d'objet long est fourni dans *ObjectString* et que rien n'est fourni dans *ObjectName*, la valeur renvoyée dans cette zone est la même que celle fournie dans *ObjectString*.

Si cette zone est omise (c'est-à-dire que *ResObjectString.VSBufSize* est égal à zéro), *ResObjectString* n'est pas renvoyé, mais la longueur est renvoyée dans *ResObjectString.VSLength*. Si la longueur est inférieure à la chaîne *ResObjectcomplète*, elle est tronquée et renvoie autant de caractères les plus à droite que la longueur indiquée.

Si *ResObjectString* est spécifié de manière incorrecte, conformément à la description de l'utilisation de la structure MQCHARV , ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie MQRC\_RES\_OBJECT\_STRING\_ERROR.

## **MQSMPO-Définition des options de propriété de message**

La structure **MQSMPO** permet aux applications de spécifier des options qui contrôlent la manière dont les propriétés des messages sont définies. La structure est un paramètre d'entrée dans l'appel **MQSETMP** .

### **Disponibilité**

Tous les systèmes IBM MQ et clients IBM MQ .

### **Jeu de caractères et codage**

Les données dans **MQSMPO** doivent être dans le jeu de caractères de l'application et dans le codage de l'application ( **MQENC\_NATIVE** ).

### **Zones**

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Tableau 531. Zones dans MQSMPO

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
StrucId (identificateur de structure)	ID_STRUC_MQSMPO_	'SMPO'
Version (numéro de version de la structure)	MQSMPO_VERSION_1	1
Options (options)	MQSMPO_NONE	0
ValueEncoding (codage de valeur de propriété)	MQENC_NATIVE	Dépend de l'environnement
ValueCCSID (jeu de caractères de la valeur de propriété)	MQCCSI_APPL	-3

**Remarques :**

1. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.
2. Dans le langage de programmation C, la variable macroMQSMPO\_DEFAULT contient les valeurs répertoriées dans la table. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQSMPO MySMPO = {MQSMPO_DEFAULT};
```

## Déclarations de langue

### Déclaration C pour MQSMPO

```
typedef struct tagMQSMPO MQSMPO;
struct tagMQSMPO {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;          /* Structure version number */
    MQLONG     Options;          /* Options that control the action of MQSETMP */
    MQLONG     ValueEncoding;    /* Encoding of Value */
    MQLONG     ValueCCSID;       /* Character set identifier of Value */
};
```

### Déclaration COBOL pour MQSMPO

```
** MQSMPO structure
10 MQSMPO.
** Structure identifier
15 MQSMPO-STRUCID PIC X(4).
** Structure version number
15 MQSMPO-VERSION PIC S9(9) BINARY.
** Options that control the action of MQSETMP
15 MQSMPO-OPTIONS PIC S9(9) BINARY.
** Encoding of VALUE
15 MQSMPO-VALUEENCODING PIC S9(9) BINARY.
** Character set identifier of VALUE
15 MQSMPO-VALUECCSID PIC S9(9) BINARY.
```

### Déclaration PL/I pour MQSMPO

```
dcl
1 MQSMPO based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 Options fixed bin(31), /* Options that control the action of MQSETMP */
```

```

3 ValueEncoding  fixed bin(31), /* Encoding of Value */
3 ValueCCSID     fixed bin(31), /* Character set identifier of Value */

```

## Déclaration High Level Assembler pour MQSMPO

```

MQSMPO          DSECT
MQSMPO_STRUCID DS  CL4  Structure identifier
MQSMPO_VERSION DS  F    Structure version number
MQSMPO_OPTIONS DS  F    Options that control the action of
*               MQSETMP
MQSMPO_VALUEENCODING DS  F    Encoding of VALUE
MQSMPO_VALUECCSID DS  F    Character set identifier of VALUE
MQSMPO_LENGTH   EQU  *-MQSMPO
MQSMPO_AREA     DS  CL(MQSMPO_LENGTH)

```

### **StrucId (MQCHAR4) pour MQSMPO**

Il s'agit de l'identificateur de structure de la structure des options de propriété de message définie. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQSMPO\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUC\_MQSMPO\_**

Identificateur de la structure des options de définition des propriétés de message.

Pour le langage de programmation C, la constante **MQSMPO\_STRUC\_ID\_ARRAY** est également définie. Il a la même valeur que **MQSMPO\_STRUC\_ID**, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQSMPO**

Il s'agit du numéro de version de la structure ; la valeur doit être:

#### **MQSMPO\_VERSION\_1**

Version-1 définit la structure des options de propriété de message.

La constante suivante indique le numéro de version de la version en cours:

#### **MQSMPO\_CURRENT\_VERSION**

Version actuelle de la structure des options de définition des propriétés de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **MQSMPO\_VERSION\_1**.

### **Options (MQLONG) pour MQSMPO**

#### **Options de localisation**

Les options suivantes se rapportent à l'emplacement relatif de la propriété par rapport au curseur de propriété:

##### **MQSMPO\_SET\_FIRST**

Définit la valeur de la première propriété qui correspond au nom spécifié ou, si elle n'existe pas, ajoute une nouvelle propriété après toutes les autres propriétés avec une hiérarchie correspondante.

##### **MQSMPO\_SET\_PROP\_UNDER\_CURSOR**

Définit la valeur de la propriété indiquée par le curseur de propriété. La propriété indiquée par le curseur de propriété est celle qui a été interrogée pour la dernière fois à l'aide de l'option MQIMPO\_INQ\_FIRST ou MQIMPO\_INQ\_NEXT.

Le curseur de propriété est réinitialisé lorsque le descripteur de message est réutilisé sur un appel MQGET ou lorsque le descripteur de message est spécifié dans la zone *MsgHandle* de la structure MQGMO ou MQPMO sur un appel MQPUT.

Si cette option est utilisée alors que le curseur de propriété n'a pas encore été établi ou si le pointeur de propriété du curseur de propriété a été supprimé, l'appel échoue avec le code achèvement MQCC\_FAILED et le code anomalie MQRC\_PROPERTY\_NOT\_AVAILABLE.

#### **MQSMPO\_SET\_PROP\_BEFORE\_CURSOR**

Définit une nouvelle propriété avant la propriété pointée par le curseur de propriété. La propriété indiquée par le curseur de propriété est celle qui a été interrogée pour la dernière fois à l'aide de l'option MQIMPO\_INQ\_FIRST ou MQIMPO\_INQ\_NEXT.

Le curseur de propriété est réinitialisé lorsque le descripteur de message est réutilisé sur un appel MQGET ou lorsque le descripteur de message est spécifié dans la zone *MsgHandle* de la structure MQGMO ou MQPMO sur un appel MQPUT.

Si cette option est utilisée alors que le curseur de propriété n'a pas encore été établi ou si le pointeur de propriété du curseur de propriété a été supprimé, l'appel échoue avec le code achèvement MQCC\_FAILED et le code anomalie MQRC\_PROPERTY\_NOT\_AVAILABLE.

#### **MQSMPO\_SET\_PROP\_AFTER\_CURSOR**

Définit une nouvelle propriété après la propriété pointée par le curseur de propriété. La propriété indiquée par le curseur de propriété est celle qui a été interrogée pour la dernière fois à l'aide de l'option MQIMPO\_INQ\_FIRST ou MQIMPO\_INQ\_NEXT.

Le curseur de propriété est réinitialisé lorsque le descripteur de message est réutilisé sur un appel MQGET ou lorsque le descripteur de message est spécifié dans la zone *MsgHandle* de la structure MQGMO ou MQPMO sur un appel MQPUT.

Si cette option est utilisée alors que le curseur de propriété n'a pas encore été établi ou si le pointeur de propriété du curseur de propriété a été supprimé, l'appel échoue avec le code achèvement MQCC\_FAILED et le code anomalie MQRC\_PROPERTY\_NOT\_AVAILABLE.

#### **MQSMPO\_XX\_ENCODE\_CASE\_ONE annexes**

Entraîne l'ajout d'une nouvelle propriété après toutes les autres propriétés avec une hiérarchie correspondante. S'il existe au moins une propriété correspondant au nom spécifié, une nouvelle propriété est ajoutée à la fin après la fin de cette liste de propriétés.

Cette option permet de créer une liste de propriétés portant le même nom.

Si vous n'avez besoin d'aucune des options décrites, utilisez l'option suivante:

#### **MQSMPO\_NONE**

Aucune option spécifiée.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQSMPO\_SET\_FIRST.

#### ***ValueEncoding (MQLONG) pour MQSMPO***

Codage de la valeur de propriété à définir si la valeur est numérique.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQENC\_NATIVE.

#### ***ValueCCSID (MQLONG) pour MQSMPO***

Jeu de caractères de la valeur de propriété à définir si la valeur est une chaîne de caractères.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQCCSI\_APPL.

### **MQSRO-Options de demande d'abonnement**

La structure MQSRO permet à l'application de spécifier des options qui contrôlent la manière dont une demande d'abonnement est effectuée. La structure est un paramètre d'entrée-sortie dans l'appel MQSUBRQ.



## Disponibilité

La structure MQSRO est disponible sur les plateformes suivantes:

-  AIX
-  IBM i
-  Linux
-  Windows
-  z/OS

et pour IBM MQ MQI clients connecté à ces systèmes.

## Version

La version actuelle de MQSRO est MQSRO\_VERSION\_1.

## Jeu de caractères et codage

Les données de MQSRO doivent se trouver dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par MQENC\_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID_STRUC_MQS	'SRO~'
<u>Version</u> (numéro de version de la structure)	MQSRO_VERSION_1	1
<u>Options</u> (options)	MQSRO_AUCUN	0
<u>NumPubs</u> (nombre de publications)	Aucun	0

### Remarques :

1. Le symbole ~ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable macroMQSRO\_DEFAULT contient les valeurs répertoriées dans la table. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQSRO MySRO = {MQSRO_DEFAULT};
```

## Déclarations de langue

Déclaration C pour MQSRO

```
typedef struct tagMQSRO MQSRO;  
struct tagMQSRO {  
    MQCHAR4   StructId;           /* Structure identifier */  
    MQLONG    Version;           /* Structure version number */
```

```

MQLONG Options; /* Options that control the action of MQSUBRQ */
MQLONG NumPubs; /* Number of publications sent */
/* Ver:1 */
};

```

## Déclaration COBOL pour MQSRO

```

** MQSRO structure
10 MQSRO.
** Structure identifier
15 MQSRO-STRUCID PIC X(4).
** Structure version number
15 MQSRO-VERSION PIC S9(9) BINARY.
** Options that control the action of MQSUBRQ
15 MQSRO-OPTIONS PIC S9(9) BINARY.
** Number of publications sent
15 MQSRO-NUMPUBS PIC S9(9) BINARY.

```

## Déclaration PL/I pour MQSRO

```

dcl
1 MQSRO based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 Options fixed bin(31), /* Options that control the action of MQSUBRQ */
3 NumPubs fixed bin(31); /* Number of publications sent */

```

## Déclaration High Level Assembler pour MQSRO

```

MQSRO DSECT
MQSRO_STRUCID DS CL4 Structure identifier
MQSRO_VERSION DS F Structure version number
MQSRO_OPTIONS DS F Options that control the action of MQSUBRQ
MQSRO_NUMPUBS DS F Number of publications sent
*
MQSRO_LENGTH EQU *-MQSRO
ORG MQSRO
MQSRO_AREA DS CL(MQSRO_LENGTH)

```

### **StrucId (MQCHAR4) pour MQSRO**

Il s'agit de l'identificateur de structure de la structure des options de demande d'abonnement. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQSRO\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUC\_MQS**

Identificateur de la structure des options de demande d'abonnement.

Pour le langage de programmation C, la constante MQSRO\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQSRO\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQSRO**

Il s'agit du numéro de version de la structure ; la valeur doit être:

#### **MQSRO\_VERSION\_1**

Structure des options de demande d'abonnement Version-1 .

La constante suivante indique le numéro de version de la version en cours:

#### **VERSION\_MQSRO\_CURRENT\_VERSION**

Version actuelle de la structure des options de demande d'abonnement.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQSRO\_VERSION\_1.

### **Options (MQLONG) pour MQSRO**

L'une des options suivantes doit être spécifiée. Vous ne pouvez indiquer qu'une seule option.

### **MQSRO\_FAIL\_IF QUIESCING**

L'appel MQSUBRQ échoue si le gestionnaire de files d'attente est à l'état de mise au repos. Sous z/OS, pour une application CICS ou IMS, cette option force également l'appel MQSUBRQ à échouer si la connexion est à l'état de mise au repos.

**Option par défaut:** Si l'option décrite précédemment n'est pas requise, l'option suivante doit être utilisée:

### **MQSRO\_AUCUN**

Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. Toutes les options sont définies sur leur valeur par défaut.

MQSRO\_NONE aide à la documentation du programme. Bien que cette option ne soit pas destinée à être utilisée avec une autre option, car sa valeur est zéro, cette utilisation ne peut pas être détectée.

### **NumPubs (MQLONG) pour MQSRO**

Il s'agit d'une zone de sortie renvoyée à l'application pour indiquer le nombre de publications envoyées à la file d'attente d'abonnement à la suite de cet appel. Bien que ce nombre de publications ait été envoyé à la suite de cet appel, rien ne garantit que ce nombre de messages sera disponible pour l'application, en particulier s'il s'agit de messages non persistants.

Il peut y avoir plusieurs publications si la rubrique abonnée contient un caractère générique. Si aucun caractère générique n'était présent dans la chaîne de rubrique lors de la création de l'abonnement représenté par *Hsub*, une seule publication au maximum est envoyée suite à cet appel.

### **MQSTS-Structure de génération de rapports de statut**

La structure MQSTS est un paramètre de sortie de la commande MQSTAT. La commande MQSTAT permet d'extraire des informations de statut. Ces informations sont renvoyées dans une structure MQSTS.

### **Jeu de caractères et codage**

Les données de type caractère dans MQSTS se trouvent dans le jeu de caractères du gestionnaire de files d'attente local ; elles sont fournies par l'attribut de gestionnaire de files d'attente *CodedCharSetId*. Les données numériques dans MQSTS sont dans le codage machine natif ; elles sont fournies par *Encoding*.

### **Zones**

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

<i>Tableau 532. Zones dans MQSTS</i>		
<b>Nom et description de la zone</b>	<b>Nom de la constante</b>	<b>Valeur initiale (le cas échéant) de la constante</b>
<u>StrucId</u> (identificateur de structure)	ID_STRUCTURE_MQST	'STAT~'
<u>Version</u> (numéro de version de la structure)	MQSTS_VERSION_1	1
<u>CompCode</u> (code achèvement de la première erreur)	MQCC_OK	0
<u>Raison</u> (code raison de la première erreur)	MQRC_NONE	0
<u>PutSuccessCount</u> (nombre d'appels d'insertion asynchrones ayant abouti)	Aucun	0

Tableau 532. Zones dans MQSTS (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>PutWarningNombre</u> (nombre d'appels d'insertion asynchrones ayant généré des avertissements)	Aucun	0
<u>PutFailureNombre</u> (nombre d'appels d'insertion asynchrones ayant échoué)	Aucun	0
<u>ObjectType</u> (type de l'objet défaillant)	MQOT_Q	1
<u>ObjectName</u> (nom de l'objet défaillant)	Aucun	Chaîne nulle ou blancs
<u>ObjectQMgrNom</u> (nom du gestionnaire de files d'attente propriétaire de l'objet défaillant)	Aucun	Chaîne nulle ou blancs
<u>ResolvedObjectName</u> (nom résolu de la file d'attente de destination)	Aucun	Chaîne nulle ou blancs
<u>ResolvedQMgrNom</u> (nom résolu du gestionnaire de files d'attente de destination)	Aucun	Chaîne nulle ou blancs
<b>Remarque :</b> Les zones restantes sont ignorées si la version est inférieure à MQSTS_VERSION_2.		
<u>ObjectString</u> (nom d'objet long de l'objet défaillant)	MQCHARV_VALEUR PAR DEFAULT	{NULL,0,0,0,-3}
<u>SubName</u> (nom de l'abonnement défaillant)	MQCHARV_VALEUR PAR DEFAULT	{NULL,0,0,0,-3}
<u>OpenOptions</u> (options d'ouverture associées à l'échec)	Aucun	0
<u>SubOptions</u> (options d'abonnement associées à l'échec)	Aucun	0
<p><b>Remarques :</b></p> <ol style="list-style-type: none"> <li>1. Le symbole ~ représente un caractère blanc unique.</li> <li>2. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.</li> <li>3. Dans le langage de programmation C, la variable de macro MQSTS_DEFAULT contient les valeurs répertoriées dans la table. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</li> </ol> <pre style="background-color: #f0f0f0; padding: 10px;">MQSTS MySTS = {MQSTS_DEFAULT};</pre>		

## Déclarations de langue

Déclaration C pour MQSTS

```
typedef struct tagMQSTS MQSTS;
struct tagMQSTS {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    CompCode;         /* Completion Code of first error */
    MQLONG    Reason;           /* Reason Code of first error */
    MQLONG    PutSuccessCount;  /* Number of Async calls succeeded */
    MQLONG    PutWarningCount; /* Number of Async calls had warnings */
    MQLONG    PutFailureCount; /* Number of Async calls had failures */
}
```

```

MQLONG   ObjectType;           /* Failing object type */
MQCHAR48  ObjectName;          /* Failing object name */
MQCHAR48  ObjectQmgrName;      /* Failing object queue manager name */
MQCHAR48  ResolvedObjectName;  /* Resolved name of destination queue */
MQCHAR48  ResolvedQmgrName;    /* Resolved name of destination qmgr */
/* Ver:1 */
MQCHARV   ObjectString;        /* Failing object long name */
MQCHARV   SubName;             /* Failing subscription name */
MQLONG    OpenOptions;         /* Failing open options */
MQLONG    SubOptions;          /* Failing subscription options */
/* Ver:2 */
};

```

## Déclaration COBOL pour MQSTS

```

** MQSTS structure
 10 MQSTS.
** Structure identifier
 15 MQSTS-STRUCID PIC X(4).
** Structure version number
 15 MQSTS-VERSION PIC S9(9) BINARY.
** Completion Code of first error
 15 MQSTS-COMPCODE PIC S9(9) BINARY.
** Reason Code of first error
 15 MQSTS-REASON PIC S9(9) BINARY.
** Number of Async put calls succeeded
 15 MQSTS-PUTSUCCESSCOUNT PIC S9(9) BINARY.
** Number of Async put calls had warnings
 15 MQSTS-PUTWARNINGCOUNT PIC S9(9) BINARY.
** Number of Async put calls had failures
 15 MQSTS-PUTFAILURECOUNT PIC S9(9) BINARY.
** Failing object type
 15 MQSTS-OBJECTTYPE PIC S9(9) BINARY.
** Failing object name
 15 MQSTS-OBJECTNAME PIC X(48).
** Failing object queue manager
 15 MQSTS-OBJECTQMGRNAME PIC X(48).
** Resolved name of destination queue
 15 MQSTS-RESOLVEDOBJECTNAME PIC X(48).
** Resolved name of destination qmgr
 15 MQSTS-RESOLVEDQMGRNAME PIC X(48).
** Ver:1 **
** Failing object long name
 15 MQSTS-OBJECTSTRING.
** Address of variable length string
 20 MQSTS-OBJECTSTRING-VSPTR POINTER.
** Offset of variable length string
 20 MQSTS-OBJECTSTRING-VSOFFSET PIC S9(9) BINARY.
** Size of buffer
 20 MQSTS-OBJECTSTRING-VSBUFSIZE PIC S9(9) BINARY.
** Length of variable length string
 20 MQSTS-OBJECTSTRING-VSLENGTH PIC S9(9) BINARY.
** CCSID of variable length string
 20 MQSTS-OBJECTSTRING-VSCCSID PIC S9(9) BINARY.
** Failing subscription name
 15 MQSTS-SUBNAME.
** Address of variable length string
 20 MQSTS-SUBNAME-VSPTR POINTER.
** Offset of variable length string
 20 MQSTS-SUBNAME-VSOFFSET PIC S9(9) BINARY.
** Size of buffer
 20 MQSTS-SUBNAME-VSBUFSIZE PIC S9(9) BINARY.
** Length of variable length string
 20 MQSTS-SUBNAME-VSLENGTH PIC S9(9) BINARY.
** CCSID of variable length string
 20 MQSTS-SUBNAME-VSCCSID PIC S9(9) BINARY.
** Failing open options
 15 MQSTS-OPENOPTIONS PIC S9(9) BINARY.
** Failing subscription options
 15 MQSTS-SUBOPTIONS PIC S9(9) BINARY.
** Ver:2 **

```

## Déclaration PL/I pour MQSTS

```

dcl
 1 MQSTS based,
 3 StrucId          char(4),          /* Structure identifier */

```

```

3 Version          fixed bin(31), /* Structure version number */
3 CompCode        fixed bin(31), /* Completion code */
3 Reason          fixed bin(31), /* Reason code */
3 PutSuccessCount fixed bin(31), /* Put success count */
3 PutWarningCount fixed bin(31), /* Put warning count */
3 PutFailureCount fixed bin(31), /* Put failure count */
3 ObjectType      fixed bin(31), /* Object type */
3 ObjectName      char(48), /* Object name */
3 ObjectQmgrName  char(48), /* Object queue manager */
3 ResolvedObjectName char(48), /* Resolved Object name */
3 ResolvedQmgrName char(48); /* Resolved Object queue manager */
/* Ver:1 */
3 ObjectString,          /* Failing object long name */
5 VSPtr pointer,        /* Address of variable length string */
5 VSOFFSET fixed bin(31), /* Offset of variable length string */
5 VSBufSize fixed bin(31), /* Size of buffer */
5 VSLength fixed bin(31), /* Length of variable length string */
5 VSCCSID fixed bin(31); /* CCSID of variable length string */
3 SubName,              /* Failing subscription name */
5 VSPtr pointer,        /* Address of variable length string */
5 VSOFFSET fixed bin(31), /* Offset of variable length string */
5 VSBufSize fixed bin(31), /* Size of buffer */
5 VSLength fixed bin(31), /* Length of variable length string */
5 VSCCSID fixed bin(31); /* CCSID of variable length string */
3 OpenOptions fixed bin(31), /* Failing open options */
3 SubOptions fixed bin(31); /* Failing subscription options */
/* Ver:2 */

```

## Déclaration High Level Assembler pour MQSTS

MQSTS	DSECT		
MQSTS_STRUCID	DS	CL4	Structure identifier
MQSTS_VERSION	DS	F	Structure version number
MQSTS_COMPCODE	DS	F	Completion code
MQSTS_REASON	DS	F	Reason code
MQSTS_PUTSUCESSCOUNT	DS	F	Success count
MQSTS_PUTWARNINGCOUNT	DS	F	Warning count
MQSTS_PUTFAILURECOUNT	DS	F	Failure count
MQSTS_OBJTYPE	DS	F	Object type
MQSTS_OBJNAME	DS	CL48	Object name
MQSTS_OBJQMGR	DS	CL48	Object queue manager
MQSTS_ROBJNAME	DS	CL48	Resolved object name
MQSTS_ROBJQMGR	DS	CL48	Resolved object queue manager
MQSTS_OBJECTSTRING	DS	0F	Force fullword alignment
MQSTS_OBJECTSTRING_VSPTR	DS	A	Address of variable length string
MQSTS_OBJECTSTRING_VSOFFSET	DS	F	Offset of variable length string
MQSTS_OBJECTSTRING_VSBUFSIZE	DS	F	Size of buffer
MQSTS_OBJECTSTRING_VSLENGTH	DS	F	Length of variable length string
MQSTS_OBJECTSTRING_VSCCSID	DS	F	CCSID of variable length string
MQSTS_OBJECTSTRING_LENGTH	EQU	*	MQSTS_OBJECTSTRING
MQSTS_OBJECTSTRING_AREA	DS		ORG MQSTS_OBJECTSTRING CL(MQSTS_OBJECTSTRING_LENGTH)
*			
MQSTS_SUBNAME	DS	0F	Force fullword alignment
MQSTS_SUBNAME_VSPTR	DS	A	Address of variable length string
MQSTS_SUBNAME_VSOFFSET	DS	F	Offset of variable length string
MQSTS_SUBNAME_VSBUFSIZE	DS	F	Size of buffer
MQSTS_SUBNAME_VSLENGTH	DS	F	Length of variable length string
MQSTS_SUBNAME_VSCCSID	DS	F	CCSID of variable length string
MQSTS_SUBNAME_LENGTH	EQ	*	MQSTS_SUBNAME
MQSTS_SUBNAME_AREA	DS		ORG MQSTS_SUBNAME CL(MQSTS_SUBNAME_LENGTH)
*			
MQSTS_OPENOPTIONS	DS	F	Failing open options
MQSTS_SUBOPTIONS	DS	F	Failing subscription option
MQSTS_LENGTH	EQU	*	MQSTS
MQSTS_AREA	DS		ORG MQSTS CL(MQSTS_LENGTH)

### Référence associée

«MQSTAT-Extraction des informations de statut», à la page 821

Utilisez l'appel MQSTAT pour extraire les informations de statut. Le type des informations de statut renvoyées est déterminé par la valeur Type indiquée dans l'appel.

### ***StrucId (MQCHAR4) pour MQSTS***

Il s'agit de l'identificateur de structure de la structure de génération de rapports de statut. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQSTS\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUCTURE\_MQST**

Identificateur de la structure de génération de rapports de statut.

Pour le langage de programmation C, la constante MQSTS\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQSTS\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### ***Version (MQLONG) pour MQSTS***

Numéro de version de la structure.

La valeur doit être:

#### **MQSTS\_VERSION\_1**

Structure de génération de rapports de statut de la version 1.

#### **MQSTS\_VERSION\_2**

Structure de génération de rapports d'état de la version 2.

La constante suivante indique le numéro de version de la version en cours:

#### **MQSTS\_CURRENT\_VERSION**

Version actuelle de la structure de génération de rapports de statut. La version actuelle est MQSTS\_VERSION\_2.

Version est toujours une zone d'entrée. Sa valeur initiale est MQSTS\_VERSION\_1.

### ***CompCode (MQLONG) pour MQSTS***

Code achèvement de l'opération faisant l'objet d'un rapport.

L'interprétation de CompCode dépend de la valeur du paramètre MQSTAT **Type** .

#### **MQSTAT\_TYPE\_ASYNC\_ERROR**

Il s'agit du code achèvement résultant d'une opération d'insertion asynchrone précédente sur l'objet spécifié dans `ObjectName`.

#### **MQSTAT\_TYPE\_RECONNEXION**

Si la connexion se reconnecte ou n'a pas pu se reconnecter, il s'agit du code achèvement qui a provoqué le début de la reconnexion.

Si la connexion est actuellement connectée, la valeur est MQCC\_OK.

#### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

Si la reconnexion a échoué, il s'agit du code achèvement qui a provoqué l'échec de la reconnexion.

Si la connexion est actuellement connectée ou reconnectée, la valeur est MQCC\_OK.

CompCode est toujours un champ de sortie. Sa valeur initiale est MQCC\_OK.

### ***Motif (MQLONG) pour MQSTS***

Code anomalie de l'opération faisant l'objet d'un rapport.

L'interprétation de Reason dépend de la valeur du paramètre MQSTAT **Type** .

#### **MQSTAT\_TYPE\_ASYNC\_ERROR**

Il s'agit du code raison résultant d'une opération d'insertion asynchrone précédente sur l'objet spécifié dans `ObjectName`.

### **MQSTAT\_TYPE\_RECONNEXION**

Si la connexion est en cours de reconnexion ou n'a pas pu être reconnectée, il s'agit du code raison qui a provoqué le début de la reconnexion.

Si la connexion est actuellement connectée, la valeur est MQRC\_NONE.

### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

Si la reconnexion a échoué, il s'agit du code raison qui a provoqué l'échec de la reconnexion.

Si la connexion est actuellement connectée ou reconnectée, la valeur est MQRC\_NONE.

Reason est un champ de sortie. Sa valeur initiale est MQRC\_NONE.

### ***PutSuccessNombre (MQLONG) pour MQSTS***

Nombre d'opérations d'insertion asynchrones ayant abouti.

La valeur de PutSuccessCount dépend de la valeur du paramètre MQSTAT **Type** .

### **MQSTAT\_TYPE\_ASYNC\_ERROR**

Nombre d'opérations d'insertion asynchrone dans l'objet nommé dans la structure MQSTS qui s'est terminée avec MQCC\_OK.

### **MQSTAT\_TYPE\_RECONNEXION**

Zéro.

### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

Zéro.

PutSuccessCount est un champ de sortie. Sa valeur initiale est zéro.

### ***PutWarningNombre (MQLONG) pour MQSTS***

Nombre d'opérations d'insertion asynchrones qui se sont terminées par un avertissement.

La valeur de PutWarningCount dépend de la valeur du paramètre MQSTAT **Type** .

### **MQSTAT\_TYPE\_ASYNC\_ERROR**

Nombre d'opérations d'insertion asynchrone dans l'objet nommé dans la structure MQSTS qui s'est terminée avec MQCC\_WARNING.

### **MQSTAT\_TYPE\_RECONNEXION**

Zéro.

### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

Zéro.

PutWarningCount est un champ de sortie. Sa valeur initiale est zéro.

### ***PutFailureNombre (MQLONG) pour MQSTS***

Nombre d'opérations d'insertion asynchrones ayant échoué.

La valeur de PutFailureCount dépend de la valeur du paramètre MQSTAT **Type** .

### **MQSTAT\_TYPE\_ASYNC\_ERROR**

Nombre d'opérations d'insertion asynchrone dans l'objet nommé dans la structure MQSTS qui s'est terminée avec MQCC\_FAILED.

### **MQSTAT\_TYPE\_RECONNEXION**

Zéro.

### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

Zéro.



PutFailureCount est un champ de sortie. Sa valeur initiale est zéro.

### **ObjectType (MQLONG) pour MQSTS**

Type de l'objet nommé dans *ObjectName* faisant l'objet d'un rapport.

Les valeurs possibles de *ObjectType* sont répertoriées dans «MQOT\_\* (types d'objet et types d'objet étendu)», à la page 166.

*ObjectType* est un champ de sortie. Sa valeur initiale est MQOT\_Q.

### **ObjectName (MQCHAR48) pour MQSTS**

Nom de l'objet faisant l'objet du rapport.

L'interprétation de *ObjectName* dépend de la valeur du paramètre MQSTAT **Type** .

#### **MQSTAT\_TYPE\_ASYNC\_ERROR**

Il s'agit du nom de la file d'attente ou de la rubrique utilisée dans l'opération d'insertion, dont l'échec est signalé dans les zones *CompCode* et *Reason* de la structure MQSTS .

#### **MQSTAT\_TYPE\_RECONNEXION**

Si la connexion est reconnectée, il s'agit du nom du gestionnaire de files d'attente associé à la connexion.

#### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

Si la reconnexion a échoué, il s'agit du nom de l'objet qui a provoqué l'échec de la reconnexion. La raison de l'échec est indiquée dans les zones *CompCode* et *Reason* de la structure MQSTS .

*ObjectName* est un champ de sortie. Sa valeur initiale est la chaîne nulle en C et 48 caractères blancs dans les autres langages de programmation.

### **ObjectQMgrNom (MQCHAR48) pour MQSTS**

Nom du gestionnaire de files d'attente faisant l'objet d'un rapport.

L'interprétation de *ObjectQMgrName* dépend de la valeur du paramètre MQSTAT **Type** .

#### **MQSTAT\_TYPE\_ASYNC\_ERROR**

Il s'agit du nom du gestionnaire de files d'attente sur lequel l'objet *ObjectName* est défini. Un nom entièrement vide jusqu'au premier caractère NULL ou jusqu'à la fin de la zone indique le gestionnaire de files d'attente auquel l'application est connectée (le gestionnaire de files d'attente local).

#### **MQSTAT\_TYPE\_RECONNEXION**



La zone **ObjectQMgrName** contient le nom d'un gestionnaire de files d'attente auquel la reconnexion est demandée ou est vide si aucun gestionnaire de files d'attente n'est spécifié. Si possible, le client tente de se reconnecter à un gestionnaire de files d'attente de ce nom.



Non renseigné.

#### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

Si la reconnexion a échoué, il s'agit du nom de l'objet qui a provoqué l'échec de la reconnexion. La raison de l'échec est indiquée dans les zones *CompCode* et *Reason* de la structure MQSTS .

*ObjectQMgrName* est un champ de sortie. Sa valeur est la chaîne nulle en C et 48 caractères blancs dans les autres langages de programmation.

### **ResolvedObjectNom (MQCHAR48) pour MQSTS**

Le nom de l'objet nommé dans *ObjectName* après que le gestionnaire de files d'attente local a résolu le nom.

L'interprétation de `ResolvedObjectName` dépend de la valeur du paramètre **MQSTAT Type** .

#### **MQSTAT\_TYPE\_ASYNC\_ERROR**

`ResolvedObjectName` est le nom de l'objet nommé dans *ObjectName* après que le gestionnaire de files d'attente local a résolu le nom. Le nom renvoyé est le nom d'un objet qui existe sur le gestionnaire de files d'attente identifié par *ResolvedQMGrName*.

#### **MQSTAT\_TYPE\_RECONNEXION**

Non renseigné.

#### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

Non renseigné.

`ResolvedObjectName` est un champ de sortie. Sa valeur initiale est la chaîne nulle en C et 48 caractères blancs dans les autres langages de programmation.

#### ***ResolvedQMGrNom (MQCHAR48) pour MQSTS***

Nom du gestionnaire de files d'attente de destination une fois que le gestionnaire de files d'attente local a résolu le nom.

L'interprétation de `ResolvedQMGrName` dépend de la valeur du paramètre **MQSTAT Type** .

#### **MQSTAT\_TYPE\_ASYNC\_ERROR**

`ResolvedQMGrName` est le nom du gestionnaire de files d'attente de destination une fois que le gestionnaire de files d'attente local a résolu le nom. Le nom renvoyé est le nom du gestionnaire de files d'attente qui possède l'objet identifié par *ResolvedObjectName*. *ResolvedQMGrName* peut être le nom du gestionnaire de files d'attente local.

#### **MQSTAT\_TYPE\_RECONNEXION**

Non renseigné.

#### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

Non renseigné.

`ResolvedQMGrName` est toujours un champ de sortie. Sa valeur initiale est la chaîne nulle en C et 48 caractères blancs dans les autres langages de programmation.

#### ***ObjectString (MQCHARV) pour MQSTS***

Nom d'objet long de l'objet défaillant signalé. Présent uniquement dans la version 2 de MQSTS ou ultérieure.

L'interprétation de `ObjectString` dépend de la valeur du paramètre **MQSTAT Type** .

#### **MQSTAT\_TYPE\_ASYNC\_ERROR**

Il s'agit du nom d'objet long de la file d'attente ou de la rubrique utilisée dans l'opération MQPUT , qui a échoué.

#### **MQSTAT\_TYPE\_RECONNEXION**

Chaîne de longueur zéro

#### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

Il s'agit du nom d'objet long de l'objet qui a provoqué l'échec de la reconnexion.

`ObjectString` est un champ de sortie. Sa valeur initiale est une chaîne de longueur nulle.

#### ***SubName (MQCHARV) pour MQSTS***

Nom de l'abonnement défaillant. Présent uniquement dans la version 2 de MQSTS ou ultérieure.

L'interprétation de `SubName` dépend de la valeur du paramètre **MQSTAT Type** .

### **MQSTAT\_TYPE\_ASYNC\_ERROR**

Chaîne de longueur zéro.

### **MQSTAT\_TYPE\_RECONNEXION**

Chaîne de longueur zéro.

### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

Nom de l'abonnement qui a provoqué l'échec de la reconnexion. Si aucun nom d'abonnement n'est disponible ou si l'échec n'est pas lié à un abonnement, il s'agit d'une chaîne de longueur nulle.

SubName est un champ de sortie. Sa valeur initiale est une chaîne de longueur nulle.

### ***OpenOptions (MQLONG) pour MQSTS***

OpenOptions utilisé pour ouvrir l'objet faisant l'objet d'un rapport. Présent uniquement dans la version 2 de MQSTS ou ultérieure.

La valeur de OpenOptions dépend de la valeur du paramètre MQSTAT **Type** .

### **MQSTAT\_TYPE\_ASYNC\_ERROR**

Zéro.

### **MQSTAT\_TYPE\_RECONNEXION**

Zéro.

### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

OpenOptions utilisé lorsque l'incident s'est produit. La raison de l'échec est indiquée dans les zones *CompCode* et *Reason* de la structure MQSTS .

OpenOptions est un champ de sortie. Sa valeur initiale est zéro.

### ***SubOptions (MQLONG) pour MQSTS***

SubOptions utilisé pour ouvrir l'abonnement défaillant. Présent uniquement dans la version 2 de MQSTS ou ultérieure.

L'interprétation de SubOptions dépend de la valeur du paramètre MQSTAT **Type** .

### **MQSTAT\_TYPE\_ASYNC\_ERROR**

Zéro.

### **MQSTAT\_TYPE\_RECONNEXION**

Zéro.

### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

SubOptions utilisé lorsque l'incident s'est produit. Si l'échec n'est pas lié à l'abonnement à une rubrique, la valeur renvoyée est zéro.

SubOptions est un champ de sortie. Sa valeur initiale est zéro.

## **MQTM-Message de déclenchement**

La structure MQTM décrit les données du message de déclenchement envoyé par le gestionnaire de files d'attente à une application de moniteur de déclenchement lorsqu'un événement de déclenchement se produit pour une file d'attente. Cette structure fait partie de l'interface TMI ( IBM MQ Trigger Monitor Interface), qui est l'une des interfaces de l'infrastructure IBM MQ .

### **Format**

MQFMT\_TRIGGER.

## Jeu de caractères et codage



Les données de type caractère dans MQTM se trouvent dans le jeu de caractères du gestionnaire de files d'attente qui génère le MQTM. Les données numériques dans MQTM se trouvent dans le codage machine du gestionnaire de files d'attente qui génère le MQTM.

Le jeu de caractères et le codage du MQTM sont fournis par les zones *CodedCharSetId* et *Encoding* dans:

- MQMD (si la structure MQTM est au début des données de message), ou
- Structure d'en-tête qui précède la structure MQTM (tous les autres cas).

## Utilisation

Une application de moniteur de déclenchement peut avoir besoin de transmettre une partie ou la totalité des informations du message de déclenchement à l'application démarrée par l'application de moniteur de déclenchement. Les informations qui peuvent être requises par l'application démarrée incluent *QName*, *TriggerData* et *UserData*. L'application de moniteur de déclenchement peut transmettre la structure MQTM directement à l'application démarrée ou transmettre une structure MQTMC2 à la place, en fonction de ce qui est autorisé par l'environnement et pratique pour l'application démarrée. Pour plus d'informations sur MQTMC2, voir «MQTMC2 -Message de déclenchement 2 (format de caractère)», à la page 634.

-  Sous z/OS, pour une application MQAT\_CICS démarrée à l'aide de la transaction CKTI, la totalité de la structure de message de déclenchement MQTM est mise à la disposition de la transaction démarrée ; les informations peuvent être extraites à l'aide de la commande EXEC CICS RETRIEVE.
-  Sous IBM i, l'application de moniteur de déclenchement fournie avec IBM MQ transmet une structure MQTMC2 à l'application démarrée.

Pour plus d'informations sur l'utilisation de déclencheurs, voir [Démarrage d'applications IBM MQ à l'aide de déclencheurs](#).

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<a href="#">StrucId</a> (identificateur de structure)	ID_STRUC_MQTM	'TM- -'
<a href="#">Version</a> (numéro de version de la structure)	MQTM_VERSION_1	1
<a href="#">QName</a> (nom de la file d'attente de déclenchement)	Aucun	Chaîne nulle ou blancs
<a href="#">ProcessName</a> (nom de l'objet de processus)	Aucun	Chaîne nulle ou blancs
<a href="#">TriggerData</a> (données de déclencheur)	Aucun	Chaîne nulle ou blancs
<a href="#">ApplType</a> (type d'application)	Aucun	0
<a href="#">AppId</a> (identificateur d'application)	Aucun	Chaîne nulle ou blancs
<a href="#">EnvData</a> (données d'environnement)	Aucun	Chaîne nulle ou blancs
<a href="#">UserData</a> (données utilisateur)	Aucun	Chaîne nulle ou blancs

Tableau 533. Zones dans MQTM pour MQTM (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<p><b>Remarques :</b></p> <ol style="list-style-type: none"> <li>1. Le symbole <code>␣</code> représente un caractère blanc unique.</li> <li>2. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.</li> <li>3. Dans le langage de programmation C, la variable macro <code>MQTM_DEFAULT</code> contient les valeurs répertoriées dans la table. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</li> </ol>		
<pre>MQTM MyTM = {MQTM_DEFAULT};</pre>		

## Déclarations de langue

### Déclaration C pour MQTM

```
typedef struct tagMQTM MQTM;
struct tagMQTM {
    MQCHAR4   StrucId;        /* Structure identifier */
    MQLONG    Version;       /* Structure version number */
    MQCHAR48  QName;        /* Name of triggered queue */
    MQCHAR48  ProcessName;   /* Name of process object */
    MQCHAR64  TriggerData;   /* Trigger data */
    MQLONG    ApplType;      /* Application type */
    MQCHAR256 ApplId;        /* Application identifier */
    MQCHAR128 EnvData;       /* Environment data */
    MQCHAR128 UserData;      /* User data */
};
```

### Déclaration COBOL pour MQTM

```
** MQTM structure
10 MQTM.
** Structure identifier
15 MQTM-STRUCID PIC X(4).
** Structure version number
15 MQTM-VERSION PIC S9(9) BINARY.
** Name of triggered queue
15 MQTM-QNAME PIC X(48).
** Name of process object
15 MQTM-PROCESSNAME PIC X(48).
** Trigger data
15 MQTM-TRIGGERDATA PIC X(64).
** Application type
15 MQTM-APPLTYPE PIC S9(9) BINARY.
** Application identifier
15 MQTM-APPLID PIC X(256).
** Environment data
15 MQTM-ENVDATA PIC X(128).
** User data
15 MQTM-USERDATA PIC X(128).
```

### Déclaration PL/I pour MQTM

```
dcl
1 MQTM based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 QName char(48), /* Name of triggered queue */
3 ProcessName char(48), /* Name of process object */
3 TriggerData char(64), /* Trigger data */
```

```

3 ApplType    fixed bin(31), /* Application type */
3 ApplId     char(256),      /* Application identifier */
3 EnvData    char(128),      /* Environment data */
3 UserData    char(128);     /* User data */

```

### Déclaration High Level Assembler pour MQTM

```

MQTM          DSECT
MQTM_STRUCID  DS    CL4      Structure identifier
MQTM_VERSION  DS    F        Structure version number
MQTM_QNAME    DS    CL48     Name of triggered queue
MQTM_PROCESSNAME DS    CL48  Name of process object
MQTM_TRIGGERDATA DS    CL64  Trigger data
MQTM_APPLTYPE DS    F        Application type
MQTM_APPLID   DS    CL256    Application identifier
MQTM_ENVDATA  DS    CL128    Environment data
MQTM_USERDATA DS    CL128    User data
*
MQTM_LENGTH   EQU    *-MQTM
              ORG    MQTM
MQTM_AREA     DS    CL(MQTM_LENGTH)

```

### Déclaration Visual Basic pour MQTM

```

Type MQTM
  StrucId    As String*4    'Structure identifier'
  Version    As Long        'Structure version number'
  QName      As String*48   'Name of triggered queue'
  ProcessName As String*48  'Name of process object'
  TriggerData As String*64  'Trigger data'
  ApplType   As Long        'Application type'
  ApplId     As String*256  'Application identifier'
  EnvData    As String*128  'Environment data'
  UserData   As String*128  'User data'
End Type

```

## MQMD pour un message de déclenchement

Tableau 534. Paramètres des zones dans le MQMD d'un message de déclenchement généré par le gestionnaire de files d'attente

Zone dans MQMD	Valeur utilisée
<i>StrucId</i>	ID_STRUCD_MQM
<i>Version</i>	MQMD_VERSION_1
<i>Report</i>	MQRO_AUCUN
<i>MsgType</i>	MQMT_DATAGRAM
<i>Expiry</i>	MQEI_UNLIMITED
<i>Feedback</i>	MQFB_AUCUN
<i>Encoding</i>	MQENC_NATIVE
<i>CodedCharSetId</i>	Attribut <b>CodedCharSetId</b> du gestionnaire de files d'attente
<i>Format</i>	MQFMT_TRIGGER
<i>Priority</i>	Attribut <b>DefPriority</b> de la file d'attente d'initialisation
<i>Persistence</i>	MQPER_NON_PERSISTENT
<i>MsgId</i>	Une valeur unique
<i>CorrelId</i>	MQCI_NONE
<i>BackoutCount</i>	0
<i>ReplyToQ</i>	Espaces vides

Tableau 534. Paramètres des zones dans le MQMD d'un message de déclenchement généré par le gestionnaire de files d'attente (suite)

Zone dans MQMD	Valeur utilisée
<i>ReplyToQMGr</i>	Nom du gestionnaire de files d'attente
<i>UserIdentifier</i>	Espaces vides
<i>AccountingToken</i>	MQACT_AUCUN
<i>AppIdentityData</i>	Espaces vides
<i>PutAppType</i>	MQAT_QMGR, ou selon le cas, pour l'agent MCA
<i>PutAppName</i>	28 premiers octets du nom du gestionnaire de files d'attente
<i>PutDate</i>	Date d'envoi du message de déclenchement
<i>PutTime</i>	Heure d'envoi du message de déclenchement
<i>AppOriginData</i>	Espaces vides

Une application qui génère un message de déclenchement est recommandée pour définir des valeurs similaires, à l'exception des suivantes:

- La zone *Priority* peut être définie sur MQPRI\_PRIORITY\_AS\_Q\_DEF (le gestionnaire de files d'attente remplace la priorité par défaut de la file d'attente d'initialisation lorsque le message est inséré).
- La zone *ReplyToQMGr* peut être mise à blanc (le gestionnaire de files d'attente change le nom du gestionnaire de files d'attente local lors de l'insertion du message).
- Définissez les zones de contexte appropriées pour l'application.

### **StrucId (MQCHAR4) pour MQTM**

Il s'agit de l'identificateur de structure de la structure de message de déclencheur. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQTM\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUC\_MQTM**

Identificateur de la structure de message de déclenchement.

Pour le langage de programmation C, la constante MQTM\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQTM\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQTM**

Il s'agit du numéro de version de la structure. La valeur doit être:

#### **MQTM\_VERSION\_1**

Numéro de version de la structure de message de déclenchement.

La constante suivante indique le numéro de version de la version en cours:

#### **MQTM\_CURRENT\_VERSION**

Version actuelle de la structure de message de déclenchement.

La valeur initiale de cette zone est MQTM\_VERSION\_1.

### **QName (MQCHAR48) pour MQTM**

Il s'agit du nom de la file d'attente pour laquelle un événement déclencheur s'est produit. Il est utilisé par l'application démarrée par l'application du moniteur de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut **QName** de la file d'attente de déclenchement ; pour plus de détails sur cet attribut, voir «Attributs des files d'attente», à la page 875.

Les noms inférieurs à la longueur définie de la zone sont complétés à droite par des blancs ; ils ne sont pas terminés prématurément par un caractère NULL.

La longueur de cette zone est indiquée par MQ\_Q\_NAME\_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

### ***ProcessName (MQCHAR48) pour MQTM***

Il s'agit du nom de l'objet de processus de gestionnaire de files d'attente spécifié pour la file d'attente de déclenchement. Il peut être utilisé par l'application du moniteur de déclenchement qui reçoit le message de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut **ProcessName** de la file d'attente identifiée par la zone *QName* ; pour plus de détails sur cet attribut, voir «Attributs des files d'attente», à la page 875.

Les noms inférieurs à la longueur définie de la zone sont toujours complétés à droite par des blancs ; ils ne sont pas terminés prématurément par un caractère NULL.

La longueur de cette zone est indiquée par MQ\_PROCESS\_NAME\_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

### ***TriggerData (MQCHAR64) pour MQTM***

Il s'agit de données au format libre à utiliser par l'application du moniteur de déclenchement qui reçoit le message de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut **TriggerData** de la file d'attente identifiée par la zone *QName* ; pour plus de détails sur cet attribut, voir «Attributs des files d'attente», à la page 875 . Le contenu de ces données n'a aucune importance pour le gestionnaire de files d'attente.

Sous z/OS, pour une application CICS démarrée à l'aide de la transaction CKTI, ces informations ne sont pas utilisées.

La longueur de cette zone est indiquée par MQ\_TRIGGER\_DATA\_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et 64 caractères blancs dans les autres langages de programmation.

### ***AppType (MQLONG) pour MQTM***

Identifie la nature du programme à démarrer et est utilisé par l'application du moniteur de déclenchement qui reçoit le message de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut **AppType** de l'objet de processus identifié par la zone *ProcessName* ; pour plus de détails sur cet attribut, voir «Attributs des définitions de processus», à la page 913 . Le contenu de ces données n'a aucune importance pour le gestionnaire de files d'attente.

*AppType* peut avoir l'une des valeurs standard suivantes. Les types définis par l'utilisateur peuvent également être utilisés, mais ils doivent être limités aux valeurs comprises entre MQAT\_USER\_FIRST et MQAT\_USER\_LAST:

#### **MQAT\_AIX**

Application AIX (même valeur que MQAT\_UNIX).

#### **MQAT\_BATCH**

application de traitement par lots

#### **MQAT\_COURTIER**

Application de courtier

#### **MQAT\_CICS**

Transaction CICS .

#### **MQAT\_CICS\_BRIDGE**

Application CICS bridge .

#### **MQAT\_CICS\_VSE**

Transaction CICS/VSE .

#### **MQAT\_DOS**

Application IBM MQ MQI client sur PC DOS.



**MQAT\_IMS**

Application IMS .

**MQAT\_IMS\_BRIDGE**

Application de pont IMS .

**MQAT\_JAVA**

Application Java .

**MQAT\_MVS**

Application MVS ou TSO (même valeur que MQAT\_ZOS).

**MQAT\_NOTES\_AGENT**

Application de l'agent Lotus Notes .

**MQAT\_OS390**

Application OS/390 (même valeur que MQAT\_ZOS).

**MQAT\_OS400**

Application IBM i .

**MQAT\_RRS\_BATCH**

Application par lots RRS.

**MQAT\_UNIX**

Application UNIX .

**MQAT\_INCONNU**

Application de type inconnu.

**Utilisateur\_MQAT**

Type d'application défini par l'utilisateur.

**MQAT\_VOS**

Application Stratus VOS.

**MQAT\_WINDOWS**

Application Windows 16 bits.

**MQAT\_WINDOWS\_NT**

Application Windows 32 bits.

**MQAT\_WLM**

Application du gestionnaire de charge de travail z/OS .

**MQAT\_XCF**

XCF.

**MQAT\_ZOS**

Application z/OS .

**MQAT\_USER\_FIRST**

Valeur la plus faible pour le type d'application défini par l'utilisateur.

**MQAT\_USER\_LAST**

Valeur la plus élevée pour le type d'application défini par l'utilisateur.

La valeur initiale de cette zone est 0.

***AppId (MQCHAR256) pour MQTM***

Il s'agit d'une chaîne de caractères qui identifie l'application à démarrer et qui est utilisée par l'application trigger-monitor qui reçoit le message de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut **AppId** de l'objet de processus identifié par la zone *ProcessName* ; pour plus de détails sur cet attribut, voir «Attributs des définitions de processus», à la page 913 . Le contenu de ces données n'a aucune importance pour le gestionnaire de files d'attente.

La signification de *AppId* est déterminée par l'application trigger-monitor. Le moniteur de déclenchement fourni par IBM MQ requiert que *AppId* soit le nom d'un programme exécutable. Les remarques suivantes s'appliquent aux environnements indiqués:

- Sous z/OS, *AppId* est:
  - Un identificateur de transaction CICS , pour les applications démarrées à l'aide de la transaction CKTI du moniteur de déclenchement CICS
  - Un identificateur de transaction IMS , pour les applications démarrées à l'aide du moniteur de déclenchement IMS CSQQTRMN
- Sur les systèmes Windows , le nom de programme peut être précédé d'un chemin d'unité et de répertoire.
- Sous IBM i, le nom de programme peut être précédé d'un nom de bibliothèque et d'un caractère /.
- Sous AIX and Linux, le nom de programme peut être précédé d'un chemin de répertoire.

La longueur de cette zone est indiquée par MQ\_PROCESS\_APPL\_ID\_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et 256 caractères blancs dans les autres langages de programmation.

### **EnvData (MQCHAR128) pour MQTM**

Il s'agit d'une chaîne de caractères qui contient des informations liées à l'environnement concernant l'application à démarrer et qui est utilisée par l'application du moniteur de déclenchement qui reçoit le message de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut **EnvData** de l'objet de processus identifié par la zone *ProcessName* ; pour plus de détails sur cet attribut, voir «[Attributs des définitions de processus](#)», à la page 913 . Le contenu de ces données n'a aucune importance pour le gestionnaire de files d'attente.

Sous z/OS, pour une application CICS démarrée à l'aide de la transaction CKTI ou une application IMS à démarrer à l'aide de la transaction CSQQTRMN, ces informations ne sont pas utilisées.

La longueur de cette zone est indiquée par MQ\_PROCESS\_ENV\_DATA\_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et 128 caractères blancs dans les autres langages de programmation.

### **UserData (MQCHAR128) pour MQTM**

Il s'agit d'une chaîne de caractères qui contient des informations utilisateur relatives à l'application à démarrer et qui est utilisée par l'application du moniteur de déclenchement qui reçoit le message de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut **UserData** de l'objet de processus identifié par la zone *ProcessName* ; pour plus de détails sur cet attribut, voir «[Attributs des définitions de processus](#)», à la page 913 . Le contenu de ces données n'a aucune importance pour le gestionnaire de files d'attente.

Pour Microsoft Windows, la chaîne de caractères ne doit pas contenir de guillemets si la définition de processus doit être transmise à **runmqtrm**.

La longueur de cette zone est indiquée par MQ\_PROCESS\_USER\_DATA\_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et 128 caractères blancs dans les autres langages de programmation.

## **MQTMC2 -Message de déclenchement 2 (format de caractère)**

Lorsqu'une application de moniteur de déclenchement extrait un message de déclenchement (MQTM) d'une file d'attente d'initialisation, il peut être nécessaire que le moniteur de déclenchement transmette tout ou partie des informations du message de déclenchement à l'application démarrée par le moniteur de déclenchement.

Les informations dont l'application démarrée peut avoir besoin incluent *QName*, *TriggerData* et *UserData*. L'application du moniteur de déclenchement peut transmettre la structure MQTM directement à l'application démarrée ou transmettre une structure MQTMC2 à la place, en fonction de ce qui est autorisé par l'environnement et pratique pour l'application démarrée.

Cette structure fait partie de l'interface TMI ( IBM MQ Trigger Monitor Interface), qui est l'une des interfaces de l'infrastructure IBM MQ .

## Jeu de caractères et codage

Les données de type caractère dans MQTMC2 se trouvent dans le jeu de caractères du gestionnaire de files d'attente local ; elles sont fournies par l'attribut de gestionnaire de files d'attente **CodedCharSetId**.

## Utilisation

La structure MQTMC2 est très similaire au format de la structure MQTM. La différence est que les zones non alphanumériques de MQTM sont remplacées dans MQTMC2 par des zones alphanumériques de même longueur et que le nom du gestionnaire de files d'attente est ajouté à la fin de la structure.

- **z/OS** Sous z/OS, pour une application MQAT\_IMS démarrée à l'aide de l'application CSQQTRMN, une structure MQTMC2 est mise à la disposition de l'application démarrée.
- **IBM i** Sous IBM i, l'application du moniteur de déclenchement fournie avec IBM MQ transmet une structure MQTMC2 à l'application démarrée.

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Tableau 535. Zones dans MQTMC2		
Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID MQTMC_STRUC_ID	'TMC↵'
<u>Version</u> (numéro de version de la structure)	MQTMC_VERSION_2	'↵↵↵2'
<u>QName</u> (nom de la file d'attente de déclenchement)	Aucun	Chaîne nulle ou blancs
<u>ProcessName</u> (nom de l'objet de processus)	Aucun	Chaîne nulle ou blancs
<u>TriggerData</u> (données de déclencheur)	Aucun	Chaîne nulle ou blancs
<u>ApplType</u> (type d'application)	Aucun	Espaces vides
<u>AppId</u> (identificateur d'application)	Aucun	Chaîne nulle ou blancs
<u>EnvData</u> (données d'environnement)	Aucun	Chaîne nulle ou blancs
<u>UserData</u> (données utilisateur)	Aucun	Chaîne nulle ou blancs
<u>QMgrName</u> (nom du gestionnaire de files d'attente)	Aucun	Chaîne nulle ou blancs
<b>Remarques :</b>		
1. Le symbole ↵ représente un caractère blanc unique.		
2. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.		
3. Dans le langage de programmation C, la variable macroMQTMC2_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:		
<pre>MQTMC2 MyTMC = {MQTMC2_DEFAULT};</pre>		

## Déclarations de langue

### Déclaration C pour MQTMC2

```
typedef struct tagMQTMC2 MQTMC2;
struct tagMQTMC2 {
    MQCHAR4    StrucId;        /* Structure identifier */
    MQCHAR4    Version;       /* Structure version number */
    MQCHAR48   QName;         /* Name of triggered queue */
    MQCHAR48   ProcessName;   /* Name of process object */
    MQCHAR64   TriggerData;   /* Trigger data */
    MQCHAR4    ApplType;      /* Application type */
    MQCHAR256  ApplId;        /* Application identifier */
    MQCHAR128  EnvData;       /* Environment data */
    MQCHAR128  UserData;      /* User data */
    MQCHAR48   QMgrName;     /* Queue manager name */
};
```

### Déclaration COBOL pour MQTMC2

```
** MQTMC2 structure
 10 MQTMC2.
** Structure identifier
 15 MQTMC2-STRUCID PIC X(4).
** Structure version number
 15 MQTMC2-VERSION PIC X(4).
** Name of triggered queue
 15 MQTMC2-QNAME PIC X(48).
** Name of process object
 15 MQTMC2-PROCESSNAME PIC X(48).
** Trigger data
 15 MQTMC2-TRIGGERDATA PIC X(64).
** Application type
 15 MQTMC2-APPLTYPE PIC X(4).
** Application identifier
 15 MQTMC2-APPLID PIC X(256).
** Environment data
 15 MQTMC2-ENVDATA PIC X(128).
** User data
 15 MQTMC2-USERSDATA PIC X(128).
** Queue manager name
 15 MQTMC2-QMGRNAME PIC X(48).
```

### Déclaration PL/I pour MQTMC2

```
dcl
 1 MQTMC2 based,
 3 StrucId char(4), /* Structure identifier */
 3 Version char(4), /* Structure version number */
 3 QName char(48), /* Name of triggered queue */
 3 ProcessName char(48), /* Name of process object */
 3 TriggerData char(64), /* Trigger data */
 3 ApplType char(4), /* Application type */
 3 ApplId char(256), /* Application identifier */
 3 EnvData char(128), /* Environment data */
 3 UserData char(128), /* User data */
 3 QMgrName char(48); /* Queue manager name */
```

### Déclaration High Level Assembler pour MQTMC2

```
MQTMC2          DSECT
MQTMC2_STRUCID  DS CL4    Structure identifier
MQTMC2_VERSION  DS CL4    Structure version number
MQTMC2_QNAME    DS CL48   Name of triggered queue
MQTMC2_PROCESSNAME DS CL48 Name of process object
MQTMC2_TRIGGERDATA DS CL64 Trigger data
MQTMC2_APPLTYPE DS CL4    Application type
MQTMC2_APPLID   DS CL256  Application identifier
MQTMC2_ENVDATA  DS CL128  Environment data
MQTMC2_USERDATA DS CL128  User data
MQTMC2_QMGRNAME DS CL48   Queue manager name
*
MQTMC2_LENGTH   EQU *-MQTMC2
```

```
MQTMC2_AREA      ORG  MQTMC2
                  DS   CL(MQTMC2_LENGTH)
```

## Déclaration Visual Basic pour MQTMC2

```
Type MQTMC2
  StrucId      As String*4   'Structure identifier'
  Version      As String*4   'Structure version number'
  QName        As String*48  'Name of triggered queue'
  ProcessName  As String*48  'Name of process object'
  TriggerData  As String*64  'Trigger data'
  ApplType     As String*4   'Application type'
  ApplId       As String*256 'Application identifier'
  EnvData      As String*128 'Environment data'
  UserData     As String*128 'User data'
  QMgrName     As String*48  'Queue manager name'
End Type
```

### **StrucId (MQCHAR4) pour MQTMC2**

Il s'agit de l'identificateur de structure du message de déclenchement 2 (format de caractères). Il s'agit toujours d'une zone d'entrée. Sa valeur est MQTMC2\_STRUC\_ID.

La valeur doit être:

#### **MQTMC2\_STRUC\_ID**

Identificateur de la structure du message de déclenchement (format de caractères).

Pour le langage de programmation C, la constante MQTMC2\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQTMC2\_STRUC\_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

### **Version (MQCHAR4) pour MQTMC2**

Numéro de version de la structure.

La valeur doit être:

#### **MQTMC\_VERSION\_2**

Structure du message de déclenchement version 2 (format de caractère).

Pour le langage de programmation C, la constante MQTMC\_VERSION\_2\_ARRAY est également définie ; elle a la même valeur que MQTMC\_VERSION\_2, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La constante suivante indique le numéro de version de la version en cours:

#### **MQTMC\_CURRENT\_VERSION**

Version en cours de la structure du message de déclenchement (format de caractères).

### **QName (MQCHAR48) pour MQTMC2**

Nom de la file d'attente déclenchée.

Voir la zone *QName* dans la structure MQTM.

### **ProcessName (MQCHAR48) pour MQTMC2**

Nom de l'objet de processus.

Voir la zone *ProcessName* dans la structure MQTM.

### **TriggerData (MQCHAR64) pour MQTMC2**

Données de déclenchement.

Voir la zone *TriggerData* dans la structure MQTM.

### ***AppType (MQCHAR4) pour MQTMC2***

Type d'application.

Cette zone contient toujours des blancs, quelle que soit la valeur de la zone *AppType* dans la structure MQTM du message de déclenchement d'origine.

### ***AppId (MQCHAR256) pour MQTMC2***

Identificateur d'application.

Voir la zone *AppId* dans la structure MQTM.

### ***EnvData (MQCHAR128) pour MQTMC2***

Données d'environnement.

Voir la zone *EnvData* dans la structure MQTM.

### ***UserData (MQCHAR128) pour MQTMC2***

Données utilisateur.

Voir la zone *UserData* dans la structure MQTM.

### ***QMGrName (MQCHAR48) pour MQTMC2***

Nom du gestionnaire de files d'attente

Il s'agit du nom du gestionnaire de files d'attente dans lequel l'événement déclencheur s'est produit.

## **MQWIH-En-tête d'informations de travail**

Si un message doit être traité par le gestionnaire de charge de travail z/OS (WLM), il doit commencer par une structure MQWIH. Cette structure décrit les informations qui doivent être présentes au début d'un message qui doit être traité par WLM.

## **Disponibilité**

Tous les systèmes IBM MQ , ainsi que les clients IBM MQ connectés à ces systèmes.

## **Format**

MQFMT\_WORK\_INFO\_HEADER.

## **Jeu de caractères et codage**

Les zones de la structure MQWIH se trouvent dans le jeu de caractères et le codage donnés par les zones *CodedCharSetId* et *Encoding* de la structure d'en-tête qui précède MQWIH, ou par les zones de la structure MQMD si MQWIH se trouve au début des données de message d'application.

Le jeu de caractères doit comporter des caractères mono-octet pour les caractères admis dans les noms de file d'attente.

## **Utilisation**

Pour toute plateforme prise en charge par IBM MQ , vous pouvez créer et transmettre un message qui inclut la structure MQWIH, mais seul un gestionnaire de files d'attente IBM MQ for z/OS peut interagir avec WLM. Par conséquent, pour que le message soit envoyé à WLM à partir d'un gestionnaire de files d'attente nonz/OS , votre réseau de gestionnaires de files d'attente doit inclure au moins un gestionnaire de files d'attente z/OS par lequel le message peut être acheminé.

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID_STRUC_MQWIH	'WIH~'
<u>Version</u> (numéro de version de la structure)	MQWIH_VERSION_1	1
<u>StrucLength</u> (longueur de la structure MQWIH)	MQWIH_LENGTH_1	120
<u>Codage</u> (codage numérique des données qui suit MQWIH)	Aucun	0
<u>CodedCharSetId</u> (identificateur de jeu de caractères des données qui suit MQWIH)	MQCCSI_UNDEFINI	0
<u>Format</u> (nom de format des données qui suivent MQWIH)	MQFMT_AUCUN	Espaces vides
<u>Indicateurs</u> (indicateurs)	MQWIH_AUCUN	0
<u>ServiceName</u> (nom du service)	Aucun	Espaces vides
<u>ServiceStep</u> (nom de l'étape de service)	Aucun	Espaces vides
<u>MsgToken</u> (jeton de message)	MQMTOK_NONE	Valeurs NULL
<u>Réservé</u> (réservé)	Aucun	Espaces vides

**Remarques :**

- Le symbole ~ représente un caractère blanc unique.
- Dans le langage de programmation C, la variable macroMQWIH\_DEFAULT contient les valeurs répertoriées dans la table. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQWIH MyWIH = {MQWIH_DEFAULT};
```

## Déclarations de langue

Déclaration C pour MQWIH

```
typedef struct tagMQWIH MQWIH;
struct tagMQWIH {
    MQCHAR4   StructId;           /* Structure identifier */
    MQLONG    Version;           /* Structure version number */
    MQLONG    StructLength;      /* Length of MQWIH structure */
    MQLONG    Encoding;         /* Numeric encoding of data that follows
    MQWIH */
    MQLONG    CodedCharSetId;    /* Character-set identifier of data that
    follows MQWIH */
    MQCHAR8   Format;           /* Format name of data that follows
    MQWIH */
    MQLONG    Flags;           /* Flags */
    MQCHAR32  ServiceName;      /* Service name */
    MQCHAR8   ServiceStep;      /* Service step name */
    MQBYTE16  MsgToken;        /* Message token */
    MQCHAR32  Reserved;        /* Reserved */
};
```

## Déclaration COBOL pour MQWIH

```
** MQWIH structure
10 MQWIH.
** Structure identifier
15 MQWIH-STRUCID PIC X(4).
** Structure version number
15 MQWIH-VERSION PIC S9(9) BINARY.
** Length of MQWIH structure
15 MQWIH-STRUCLength PIC S9(9) BINARY.
** Numeric encoding of data that follows MQWIH
15 MQWIH-ENCODING PIC S9(9) BINARY.
** Character-set identifier of data that follows MQWIH
15 MQWIH-CODEDCHARSETID PIC S9(9) BINARY.
** Format name of data that follows MQWIH
15 MQWIH-FORMAT PIC X(8).
** Flags
15 MQWIH-FLAGS PIC S9(9) BINARY.
** Service name
15 MQWIH-SERVICENAME PIC X(32).
** Service step name
15 MQWIH-SERVICESTEP PIC X(8).
** Message token
15 MQWIH-MSGTOKEN PIC X(16).
** Reserved
15 MQWIH-RESERVED PIC X(32).
```

## Déclaration PL/I pour MQWIH

```
dcl
1 MQWIH based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 StrucLength fixed bin(31), /* Length of MQWIH structure */
3 Encoding fixed bin(31), /* Numeric encoding of data that
follows MQWIH */
3 CodedCharSetId fixed bin(31), /* Character-set identifier of data
that follows MQWIH */
3 Format char(8), /* Format name of data that follows
MQWIH */
3 Flags fixed bin(31), /* Flags */
3 ServiceName char(32), /* Service name */
3 ServiceStep char(8), /* Service step name */
3 MsgToken char(16), /* Message token */
3 Reserved char(32); /* Reserved */
```

## Déclaration High Level Assembler pour MQWIH

```
MQWIH          DSECT
MQWIH_STRUCID  DS   CL4  Structure identifier
MQWIH_VERSION  DS   F    Structure version number
MQWIH_STRUCLength DS   F    Length of MQWIH structure
MQWIH_ENCODING DS   F    Numeric encoding of data that follows
*              MQWIH
MQWIH_CODEDCHARSETID DS   F    Character-set identifier of data that
*              follows MQWIH
MQWIH_FORMAT   DS   CL8  Format name of data that follows MQWIH
MQWIH_FLAGS    DS   F    Flags
MQWIH_SERVICENAME DS   CL32 Service name
MQWIH_SERVICESTEP DS   CL8  Service step name
MQWIH_MSGTOKEN DS   XL16 Message token
MQWIH_RESERVED DS   CL32 Reserved
*
MQWIH_LENGTH   EQU   *-MQWIH
               ORG   MQWIH
MQWIH_AREA     DS   CL(MQWIH_LENGTH)
```

## Déclaration Visual Basic pour MQWIH

```
Type MQWIH
  StrucId      As String*4 'Structure identifier'
  Version     As Long      'Structure version number'
  StrucLength  As Long      'Length of MQWIH structure'
```



Encoding	As Long	'Numeric encoding of data that follows' 'MQWIH'
CodedCharSetId	As Long	'Character-set identifier of data that' 'follows MQWIH'
Format	As String*8	'Format name of data that follows MQWIH'
Flags	As Long	'Flags'
ServiceName	As String*32	'Service name'
ServiceStep	As String*8	'Service step name'
MsgToken	As MQBYTE16	'Message token'
Reserved	As String*32	'Reserved'
End Type		

### ***StrucId (MQCHAR4) pour MQWIH***

Il s'agit de l'identificateur de structure de la structure d'en-tête des informations de travail. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQWIH\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUC\_MQWIH**

Identificateur de la structure d'en-tête des informations de travail.

Pour le langage de programmation C, la constante MQWIH\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQWIH\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### ***Version (MQLONG) pour MQWIH***

Il s'agit du numéro de version de la structure. La valeur doit être:

#### **MQWIH\_VERSION\_1**

Structure d'en-tête des informations de travail Version-1 .

La constante suivante indique le numéro de version de la version en cours:

#### **MQWIH\_CURRENT\_VERSION**

Version actuelle de la structure d'en-tête des informations de travail.

La valeur initiale de cette zone est MQWIH\_VERSION\_1.

### ***StrucLength (MQLONG) pour MQWIH***

Longueur de la structure MQWIH. La valeur doit être:

#### **MQWIH\_LENGTH\_1**

Longueur de la structure d'en-tête des informations de travail version-1 .

La constante suivante indique la longueur de la version en cours:

#### **MQWIH\_LONGUEUR\_EN\_COURS**

Longueur de la version actuelle de la structure d'en-tête des informations de travail.

La valeur initiale de cette zone est MQWIH\_LENGTH\_1.

### ***Codage (MQLONG) pour MQWIH***

Indique le codage numérique des données qui suivent la structure MQWIH ; il ne s'applique pas aux données numériques de la structure MQWIH elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données.

La valeur initiale de cette zone est 0.

### ***CodedCharSetId (MQLONG) pour MQWIH***

Indique l'identificateur de jeu de caractères des données qui suivent la structure MQWIH ; il ne s'applique pas aux données de type caractères de la structure MQWIH elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Vous pouvez utiliser la valeur spéciale suivante:

### **MQCCSI\_HÉRITER**

Les données de type caractères dans les données *suivant* cette structure se trouvent dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, la valeur MQCCSI\_INHERIT n'est pas renvoyée par l'appel MQGET.

MQCCSI\_INHERIT ne peut pas être utilisé si la valeur de la zone *PutApplType* dans MQMD est MQAT\_BROKER.

La valeur initiale de cette zone est MQCCSI\_UNDEFINED.

### **Format (MQCHAR8) pour MQWIH**

Indique le nom de format des données qui suivent la structure MQWIH.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de la zone *Format* dans MQMD.

La longueur de cette zone est indiquée par MQ\_FORMAT\_LENGTH. La valeur initiale de cette zone est MQFMT\_NONE.

### **Indicateurs (MQLONG) pour MQWIH**

La valeur doit être:

#### **MQWIH\_AUCUN**

Aucun indicateur.

La valeur initiale de cette zone est MQWIH\_NONE.

### **ServiceName (MQCHAR32) pour MQWIH**

Il s'agit du nom du service qui doit traiter le message.

La longueur de cette zone est indiquée par MQ\_SERVICE\_NAME\_LENGTH. La valeur initiale de cette zone est de 32 caractères blancs.

### **ServiceStep (MQCHAR8) pour MQWIH**

Il s'agit du nom de l'étape de *ServiceName* à laquelle le message est lié.

La longueur de cette zone est indiquée par MQ\_SERVICE\_STEP\_LENGTH. La valeur initiale de cette zone est de 8 caractères blancs.

### **MsgToken (MQBYTE16) pour MQWIH**

Il s'agit d'un jeton de message qui identifie le message de manière unique.

Pour les appels MQPUT et MQPUT1 , cette zone est ignorée. La longueur de cette zone est indiquée par MQ\_MSG\_TOKEN\_LENGTH. La valeur initiale de cette zone est MQMTOK\_NONE.

### **Réservé (MQCHAR32) pour MQWIH**

Il s'agit d'une zone réservée ; elle doit être vide.

## **MQXP-Bloc de paramètres d'exit**

La structure MQXP est utilisée comme paramètre d'entrée/sortie pour l'exit de croisement d'API. Pour plus d'informations sur cet exit, voir [L'exit de croisement d'API](#).

## Jeu de caractères et codage

Les données de type caractère dans MQXP se trouvent dans le jeu de caractères du gestionnaire de files d'attente local ; elles sont fournies par l'attribut de gestionnaire de files d'attente **CodedCharSetId** . Les données numériques dans MQXP sont dans le codage de la machine native ; elles sont fournies par MQENC\_NATIVE.

## Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Tableau 537. Zones dans MQXP	
Nom et description de la zone	Nom de la constante
StrucId (identificateur de structure)	ID_STRUCT_MQXP
Version (numéro de version de la structure)	MQXP_VERSION_1
ExitId (identificateur de l'exit)	EXIT MQXT_API_CROSSING_EXIT
ExitReason (raison de l'appel de l'exit)	Aucun
ExitResponse (réponse de l'exit)	Aucun
ExitCommand (code d'appel d'API)	Aucun
ExitParmCount (nombre de paramètres)	Aucun
Réservé (réservé)	Aucun
ZoneExitUser (zone utilisateur)	Aucun

## Déclarations de langue

Déclaration C pour MQXP

```
typedef struct tagMQXP MQXP;
struct tagMQXP {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    ExitId;           /* Exit identifier */
    MQLONG    ExitReason;       /* Reason for invocation of exit */
    MQLONG    ExitResponse;     /* Response from exit */
    MQLONG    ExitCommand;      /* API call code */
    MQLONG    ExitParmCount;    /* Parameter count */
    MQLONG    Reserved;         /* Reserved */
    MQBYTE16  ExitUserArea;     /* User area */
};
```

Déclaration COBOL pour MQXP

```
** MQXP structure
10 MQXP.
** Structure identifier
15 MQXP-STRUCID PIC X(4).
** Structure version number
15 MQXP-VERSION PIC S9(9) BINARY.
** Exit identifier
15 MQXP-EXITID PIC S9(9) BINARY.
** Reason for invocation of exit
15 MQXP-EXITREASON PIC S9(9) BINARY.
** Response from exit
15 MQXP-EXITRESPONSE PIC S9(9) BINARY.
** API call code
```

```

15 MQXP-EXITCOMMAND PIC S9(9) BINARY.
**   Parameter count
15 MQXP-EXITPARMCOUNT PIC S9(9) BINARY.
**   Reserved
15 MQXP-RESERVED PIC S9(9) BINARY.
**   User area
15 MQXP-EXITUSERAREA PIC X(16).

```

## Déclaration PL/I pour MQXP

```

dcl
  1 MQXP based,
  3 StrucId      char(4),          /* Structure identifier */
  3 Version     fixed bin(31),    /* Structure version number */
  3 ExitId      fixed bin(31),    /* Exit identifier */
  3 ExitReason  fixed bin(31),    /* Reason for invocation of exit */
  3 ExitResponse fixed bin(31),   /* Response from exit */
  3 ExitCommand fixed bin(31),    /* API call code */
  3 ExitParmCount fixed bin(31), /* Parameter count */
  3 Reserved    fixed bin(31),    /* Reserved */
  3 ExitUserArea char(16);        /* User area */

```

## Déclaration High Level Assembler pour MQXP

```

MQXP          DSECT
MQXP_STRUCID  DS   CL4   Structure identifier
MQXP_VERSION  DS   F     Structure version number
MQXP_EXITID   DS   F     Exit identifier
MQXP_EXITREASON DS   F   Reason for invocation of exit
MQXP_EXITRESPONSE DS   F Response from exit
MQXP_EXITCOMMAND DS   F  API call code
MQXP_EXITPARMCOUNT DS   F Parameter count
MQXP_RESERVED DS   F    Reserved
MQXP_EXITUSERAREA DS  XL16 User area
*
MQXP_LENGTH   EQU   *-MQXP
              ORG   MQXP
MQXP_AREA     DS   CL(MQXP_LENGTH)

```

### **StrucId (MQCHAR4) pour MQXP**

Il s'agit de l'identificateur de structure de la structure de paramètre d'exit. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQXP\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUCT\_MQXP**

Identificateur de la structure des paramètres d'exit.

Pour le langage de programmation C, la constante MQXP\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQXP\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQXP**

Il s'agit du numéro de version de la structure. La valeur doit être:

#### **MQXP\_VERSION\_1**

Numéro de version de la structure de bloc de paramètres d'exit.

**Remarque :** Lorsqu'une nouvelle version de cette structure est introduite, la présentation de la partie existante n'est pas modifiée. L'exit doit donc vérifier que le numéro de version est égal ou supérieur à la version la plus basse qui contient les zones que l'exit doit utiliser.

Il s'agit d'une zone d'entrée de l'exit.

### **ExitId (MQLONG) pour MQXP**

Cette valeur est définie à l'entrée de la routine d'exit et indique le type d'exit:

#### **EXIT MQXT\_API\_CROSSING\_EXIT**

Exit de croisement d'API pour CICS.

Il s'agit d'une zone d'entrée de l'exit.

### ***ExitReason (MQLONG) pour MQXP***

Ce paramètre est défini à l'entrée de la routine d'exit. Pour l'exit de croisement d'API, il indique si la routine est appelée avant ou après l'exécution de l'appel d'API:

#### **MQXR\_AVANT**

Avant l'exécution de l'API.

#### **MQXR\_APRES**

Après l'exécution de l'API.

Il s'agit d'une zone d'entrée de l'exit.

### ***ExitResponse (MQLONG) pour MQXP***

La valeur est définie par l'exit pour communiquer avec l'appelant. Les valeurs suivantes sont définies :

#### **MQXCC\_OK**

La sortie a abouti.

#### **MQXCC\_SUPPRESS\_FUNCTION**

Supprimer la fonction.

Lorsque cette valeur est définie par un exit de croisement d'API appelé *avant* l'appel d'API, l'appel d'API n'est pas effectué. Le paramètre *CompCode* de l'appel est défini sur MQCC\_FAILED, le paramètre *Reason* est défini sur MQRC\_SUPPRESSED\_BY\_EXIT et tous les autres paramètres restent tels que l'exit les a laissés.

Lorsque cette valeur est définie par un exit de croisement d'API appelé *après* l'appel d'API, elle est ignorée par le gestionnaire de files d'attente.

#### **MQXCC\_LISTE\_FONCTION**

Ignorer la fonction.

Lorsque cette valeur est définie par un exit de croisement d'API appelé *avant* l'appel d'API, l'appel d'API n'est pas effectué ; les paramètres *CompCode* et *Reason* et tous les autres paramètres restent tels que l'exit les a laissés.

Lorsque cette valeur est définie par un exit de croisement d'API appelé *après* l'appel d'API, elle est ignorée par le gestionnaire de files d'attente.

Il s'agit d'une zone de sortie de l'exit.

### ***ExitCommand (MQLONG) pour MQXP***

Cette zone est définie à l'entrée de la routine d'exit. Il identifie l'appel d'API qui a provoqué l'appel de l'exit:

#### **MQXC\_RAPPEL**

Appel CALLBACK.

#### **MQXC\_MQBACK**

Appel MQBACK.

#### **MQXC\_MQCB**

Appel MQCB.

#### **MQXC\_MQCLOSE**

Appel MQCLOSE.

#### **MQXC\_MQCMIT**

Appel MQCMIT.

#### **MQXC\_MQCTL**

Appel MQCTL.

**MQXC\_MQGET**

Appel MQGET.

**MQXC\_MQINQ**

Appel MQINQ.

**MQXC\_MQOPEN**

Appel MQOPEN.

**MQXC\_MQPUT**

Appel MQPUT.

**MQXC\_MQPUT1**

L'appel MQPUT1 .

**MQXC\_MQSET**

Appel MQSET.

**MQXC\_MQSTAT**

Appel MQSTAT.

**MQXC\_MQSUB**

Appel MQSUB.

**MQXC\_MQSUBRQ**

Appel MQSUBRQ.

Il s'agit d'une zone d'entrée de l'exit.

**ExitParmNombre (MQLONG) pour MQXP**

Cette zone est définie à l'entrée de la routine d'exit. Il contient le nombre de paramètres pris par l'appel MQ .

Tableau 538. Nombre de paramètres pour chaque appel MQ

Nom de l'appel	Nombre de paramètres
MQBACK	3
MQCLOSE	5
MQCMIT	3
MQGET	9
MQINQ	10
MQOPEN	6
MQPUT	8
MQPUT1	8
MQSET	10

Il s'agit d'une zone d'entrée de l'exit.

**Réservé (MQLONG) pour MQXP**

Il s'agit d'une zone réservée. Sa valeur n'est pas significative pour l'exit.

**Zone ExitUser(MQBYTE16) pour MQXP**

Il s'agit d'une zone disponible que l'exit peut utiliser. Il est initialisé à zéro binaire pour la longueur de la zone avant le premier appel de l'exit pour la tâche, puis toutes les modifications apportées à cette zone par l'exit sont conservées lors des appels de l'exit. La valeur suivante est définie:

**MQXUA\_NONE**

Aucune information utilisateur.

La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQXUA\_NONE\_ARRAY est également définie ; elle a la même valeur que MQXUA\_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La longueur de cette zone est indiquée par MQ\_EXIT\_USER\_AREA\_LENGTH. Il s'agit d'une zone d'entrée-sortie pour l'exit.

## MQXQH-en-tête de file d'attente de transmission

La structure MQXQH décrit les informations qui sont préfixées aux données de message d'application des messages lorsqu'ils se trouvent dans des files d'attente de transmission. Une file d'attente de transmission est un type spécial de file d'attente locale qui contient temporairement des messages destinés à des files d'attente éloignées (c'est-à-dire destinées à des files d'attente qui n'appartiennent pas au gestionnaire de files d'attente local). Une file d'attente de transmission est désignée par l'attribut de file d'attente **Usage** ayant la valeur MQUS\_TRANSMISSION.

### Format

MQFMT\_XMIT\_Q\_HEADER

### Jeu de caractères et codage

Les données de MQXQH doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par MQENC\_NATIVE.

Définissez le jeu de caractères et le codage du MQXQH dans les zones *CodedCharSetId* et *Encoding* dans :

- Le MQMD distinct (si la structure MQXQH est au début des données de message), ou
- Structure d'en-tête qui précède la structure MQXQH (tous les autres cas).

### Zones

**Remarque :** Dans le tableau suivant, les zones sont regroupées par utilisation plutôt que par ordre alphabétique. Les rubriques enfant suivent la même séquence.

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<u>StrucId</u> (identificateur de structure)	ID_STRUC_MQXQH_	' XQH- '
<u>Version</u> (numéro de version de la structure)	MQXQH_VERSION_1	1
<u>RemoteQName</u> (nom de la file d'attente de destination)	Aucun	Chaîne nulle ou blancs
<u>RemoteQMgrNom</u> (nom du gestionnaire de files d'attente de destination)	Aucun	Chaîne nulle ou blancs
<u>MsgDesc</u> (descripteur de message d'origine)	Mêmes noms et valeurs que MQMD ; voir <a href="#">Tableau 500</a> , à la page 440	-

Tableau 539. Zones de MQXQH pour MQXQH (suite)

Nom et description de la zone	Nom de la constante	Valeur initiale (le cas échéant) de la constante
<p><b>Remarques :</b></p> <ol style="list-style-type: none"> <li>1. Le symbole <code>␣</code> représente un caractère blanc unique.</li> <li>2. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.</li> <li>3. Dans le langage de programmation C, la variable macro <code>MQXQH_DEFAULT</code> contient les valeurs répertoriées dans la table. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:</li> </ol>		
<pre>MQXQH MyXQH = {MQXQH_DEFAULT};</pre>		

## Déclarations de langue

### Déclaration C pour MQXQH

```
typedef struct tagMQXQH MQXQH;
struct tagMQXQH {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;          /* Structure version number */
    MQCHAR48   RemoteQName;      /* Name of destination queue */
    MQCHAR48   RemoteQMgrName;   /* Name of destination queue manager */
    MQMD1      MsgDesc;          /* Original message descriptor */
};
```

### Déclaration COBOL pour MQXQH

```
** MQXQH structure
10 MQXQH.
** Structure identifier
15 MQXQH-STRUCID          PIC X(4).
** Structure version number
15 MQXQH-VERSION         PIC S9(9) BINARY.
** Name of destination queue
15 MQXQH-REMOTEQNAME     PIC X(48).
** Name of destination queue manager
15 MQXQH-REMOTEQMGRNAME  PIC X(48).
** Original message descriptor
15 MQXQH-MSGDESC.
** Structure identifier
20 MQXQH-MSGDESC-STRUCID  PIC X(4).
** Structure version number
20 MQXQH-MSGDESC-VERSION  PIC S9(9) BINARY.
** Report options
20 MQXQH-MSGDESC-REPORT   PIC S9(9) BINARY.
** Message type
20 MQXQH-MSGDESC-MSGTYPE  PIC S9(9) BINARY.
** Expiry time
20 MQXQH-MSGDESC-EXPIRY   PIC S9(9) BINARY.
** Feedback or reason code
20 MQXQH-MSGDESC-FEEDBACK PIC S9(9) BINARY.
** Numeric encoding of message data
20 MQXQH-MSGDESC-ENCODING PIC S9(9) BINARY.
** Character set identifier of message data
20 MQXQH-MSGDESC-CODEDCHARSETID PIC S9(9) BINARY.
** Format name of message data
20 MQXQH-MSGDESC-FORMAT   PIC X(8).
** Message priority
20 MQXQH-MSGDESC-PRIORITY PIC S9(9) BINARY.
** Message persistence
20 MQXQH-MSGDESC-PERSISTENCE PIC S9(9) BINARY.
** Message identifier
20 MQXQH-MSGDESC-MSGID    PIC X(24).
```



```

**      Correlation identifier
20 MQXQH-MSGDESC-CORRELID      PIC X(24).
**      Backout counter
20 MQXQH-MSGDESC-BACKOUTCOUNT PIC S9(9) BINARY.
**      Name of reply-to queue
20 MQXQH-MSGDESC-REPLYTOQ      PIC X(48).
**      Name of reply queue manager
20 MQXQH-MSGDESC-REPLYTOQMGR   PIC X(48).
**      User identifier
20 MQXQH-MSGDESC-USERIDENTIFIER PIC X(12).
**      Accounting token
20 MQXQH-MSGDESC-ACCOUNTINGTOKEN PIC X(32).
**      Application data relating to identity
20 MQXQH-MSGDESC-APPLIDENTITYDATA PIC X(32).
**      Type of application that put the message
20 MQXQH-MSGDESC-PUTAPPLTYPE   PIC S9(9) BINARY.
**      Name of application that put the message
20 MQXQH-MSGDESC-PUTAPPLNAME   PIC X(28).
**      Date when message was put
20 MQXQH-MSGDESC-PUTDATE      PIC X(8).
**      Time when message was put
20 MQXQH-MSGDESC-PUTTIME      PIC X(8).
**      Application data relating to origin
20 MQXQH-MSGDESC-APPLORIGINDATA PIC X(4).

```

### Déclaration PL/I pour MQXQH

```

dcl
1 MQXQH based,
3 StrucId      char(4),          /* Structure identifier */
3 Version      fixed bin(31),    /* Structure version number */
3 RemoteQName  char(48),         /* Name of destination queue */
3 RemoteQMgrName char(48),      /* Name of destination queue
                                manager */
3 MsgDesc,     /* Original message descriptor */
5 StrucId      char(4),          /* Structure identifier */
5 Version      fixed bin(31),    /* Structure version number */
5 Report       fixed bin(31),    /* Report options */
5 MsgType      fixed bin(31),    /* Message type */
5 Expiry       fixed bin(31),    /* Expiry time */
5 Feedback     fixed bin(31),    /* Feedback or reason code */
5 Encoding     fixed bin(31),    /* Numeric encoding of message
                                data */
5 CodedCharSetId fixed bin(31), /* Character set identifier of
                                message data */
5 Format        char(8),          /* Format name of message data */
5 Priority      fixed bin(31),    /* Message priority */
5 Persistence  fixed bin(31),    /* Message persistence */
5 MsgId        char(24),         /* Message identifier */
5 CorrelId     char(24),         /* Correlation identifier */
5 BackoutCount fixed bin(31),    /* Backout counter */
5 ReplyToQ     char(48),         /* Name of reply-to queue */
5 ReplyToQMgr  char(48),         /* Name of reply queue manager */
5 UserIdentifier char(12),       /* User identifier */
5 AccountingToken char(32),     /* Accounting token */
5 ApplIdentityData char(32),    /* Application data relating to
                                identity */
5 PutApplType  fixed bin(31),    /* Type of application that put the
                                message */
5 PutApplName  char(28),         /* Name of application that put the
                                message */
5 PutDate      char(8),          /* Date when message was put */
5 PutTime      char(8),          /* Time when message was put */
5 ApplOriginData char(4);       /* Application data relating to
                                origin */

```

### Déclaration High Level Assembler pour MQXQH

```

MQXQH          DSECT
MQXQH_STRUCID  DS    CL4  Structure identifier
MQXQH_VERSION  DS    F    Structure version number
MQXQH_REMOTENAME DS    CL48 Name of destination queue
MQXQH_REMOTEMGRNAME DS    CL48 Name of destination queue
                                manager
*
MQXQH_MSGDESC  DS    0F   Forcé fullword alignment
MQXQH_MSGDESC_STRUCID DS    CL4  Structure identifier
MQXQH_MSGDESC_VERSION DS    F    Structure version number

```

MQXQH_MSGDESC_REPORT	DS	F	Report options
MQXQH_MSGDESC_MSGTYPE	DS	F	Message type
MQXQH_MSGDESC_EXPIRY	DS	F	Expiry time
MQXQH_MSGDESC_FEEDBACK	DS	F	Feedback or reason code
MQXQH_MSGDESC_ENCODING	DS	F	Numeric encoding of message data
*			
MQXQH_MSGDESC_CODEDCHARSETID	DS	F	Character set identifier of message data
*			
MQXQH_MSGDESC_FORMAT	DS	CL8	Format name of message data
MQXQH_MSGDESC_PRIORITY	DS	F	Message priority
MQXQH_MSGDESC_PERSISTENCE	DS	F	Message persistence
MQXQH_MSGDESC_MSGID	DS	XL24	Message identifier
MQXQH_MSGDESC_CORRELID	DS	XL24	Correlation identifier
MQXQH_MSGDESC_BACKOUTCOUNT	DS	F	Backout counter
MQXQH_MSGDESC_REPLYTOQ	DS	CL48	Name of reply-to queue
MQXQH_MSGDESC_REPLYTOQMGR	DS	CL48	Name of reply queue manager
MQXQH_MSGDESC_USERIDENTIFIER	DS	CL12	User identifier
MQXQH_MSGDESC_ACCOUNTINGTOKEN	DS	XL32	Accounting token
MQXQH_MSGDESC_APPLIDENTITYDATA	DS	CL32	Application data relating to identity
*			
MQXQH_MSGDESC_PUTAPPLTYPE	DS	F	Type of application that put the message
*			
MQXQH_MSGDESC_PUTAPPLNAME	DS	CL28	Name of application that put the message
*			
MQXQH_MSGDESC_PUTDATE	DS	CL8	Date when message was put
MQXQH_MSGDESC_PUTTIME	DS	CL8	Time when message was put
MQXQH_MSGDESC_APPLORIGINDATA	DS	CL4	Application data relating to origin
*			
MQXQH_MSGDESC_LENGTH	EQU	*-MQXQH_MSGDESC	
	ORG	MQXQH_MSGDESC	
MQXQH_MSGDESC_AREA	DS	CL(MQXQH_MSGDESC_LENGTH)	
*			
MQXQH_LENGTH	EQU	*-MQXQH	
	ORG	MQXQH	
MQXQH_AREA	DS	CL(MQXQH_LENGTH)	

## Déclaration Visual Basic pour MQXQH

```

Type MQXQH
  StrucId      As String*4  'Structure identifier'
  Version      As Long      'Structure version number'
  RemoteQName  As String*48 'Name of destination queue'
  RemoteQMgrName As String*48 'Name of destination queue manager'
  MsgDesc      As MQMD1    'Original message descriptor'
End Type

```

## Zones du descripteur de message distinct

Un message qui se trouve dans une file d'attente de transmission comporte *deux* descripteurs de message:

- Un descripteur de message est stocké séparément des données de message ; il est appelé *descripteur de message distinct* et est généré par le gestionnaire de files d'attente lorsque le message est placé dans la file d'attente de transmission. Certaines zones du descripteur de message distinct sont copiées à partir du descripteur de message fourni par l'application sur l'appel MQPUT ou MQPUT1 .

Le descripteur de message distinct est celui qui est renvoyé à l'application dans le paramètre **MsgDesc** de l'appel MQGET lorsque le message est supprimé de la file d'attente de transmission.

- Un deuxième descripteur de message est stocké dans la structure MQXQH dans le cadre des données de message ; il est appelé *descripteur de message imbriqué* et il s'agit d'une copie du descripteur de message fourni par l'application sur l'appel MQPUT ou MQPUT1 (avec des variations mineures).

Le descripteur de message imbriqué est toujours un MQMD version-1 . Si le message inséré par l'application comporte des valeurs autres que celles par défaut pour une ou plusieurs des zones version-2 du MQMD, une structure MQMDE suit le MQXQH et est à son tour suivie des données du message d'application (le cas échéant). Le MQMDE est l'un des suivants:

- Généré par le gestionnaire de files d'attente (si l'application utilise un MQMD version-2 pour insérer le message), ou

- Déjà présent au début des données de message d'application (si l'application utilise un MQMD version-1 pour insérer le message).

Le descripteur de message imbriqué est celui qui est renvoyé à l'application dans le paramètre **MsgDesc** de l'appel MQGET lorsque le message est supprimé de la file d'attente de destination finale.

Les zones du descripteur de message distinct sont définies par le gestionnaire de files d'attente comme indiqué. Si le gestionnaire de files d'attente ne prend pas en charge le MQMD version-2 , un MQMD version-1 est utilisé sans perte de fonction.

Tableau 540. Valeurs utilisées pour les zones dans le MQMD distinct

Zone dans MQMD distinct	Valeur utilisée
<i>StrucId</i>	ID_STRUCD_MQM
<i>Version</i>	MQMD_VERSION_2
<i>Report</i>	Copié à partir du descripteur de message imbriqué, mais avec les bits identifiés par MQRO_ACCEPT_UNSUP_IF_XMIT_MASK définis sur zéro. (Cela empêche la génération d'un message de rapport COA ou COD lorsqu'un message est placé dans ou supprimé d'une file d'attente de transmission.)
<i>MsgType</i>	Copié à partir du descripteur de message imbriqué.
<i>Expiry</i>	Copié à partir du descripteur de message imbriqué.
<i>Feedback</i>	Copié à partir du descripteur de message imbriqué.
<i>Encoding</i>	MQENC_NATIVE (voir la remarque)
<i>CodedCharSetId</i>	Attribut <b>CodedCharSetId</b> du gestionnaire de files d'attente.
<i>Format</i>	MQFMT_XMIT_Q_HEADER
<i>Priority</i>	Copié à partir du descripteur de message imbriqué.
<i>Persistence</i>	Copié à partir du descripteur de message imbriqué.
<i>MsgId</i>	Une nouvelle valeur est générée par le gestionnaire de files d'attente. Cet identificateur de message est différent du <i>MsgId</i> que le gestionnaire de files d'attente a peut-être généré pour le descripteur de message intégré décrit précédemment.
<i>CorrelId</i>	<i>MsgId</i> à partir du descripteur de message imbriqué. Pour les messages insérés dans SYSTEM.CLUSTER.TRANSMIT.QUEUE, <i>CorrelId</i> est réservé à une utilisation interne.
<i>BackoutCount</i>	0
<i>ReplyToQ</i>	Copié à partir du descripteur de message imbriqué.
<i>ReplyToQMGr</i>	Copié à partir du descripteur de message imbriqué.
<i>UserIdentifier</i>	Copié à partir du descripteur de message imbriqué.
<i>AccountingToken</i>	Copié à partir du descripteur de message imbriqué. Pour les messages insérés dans SYSTEM.CLUSTER.TRANSMIT.QUEUE, <i>AccountingToken</i> est réservé à une utilisation interne.
<i>AppIdentityData</i>	Copié à partir du descripteur de message imbriqué.
<i>PutAppType</i>	MQAT_QMGR
<i>PutAppName</i>	28 premiers octets du nom du gestionnaire de files d'attente.
<i>PutDate</i>	Date à laquelle le message a été inséré dans la file d'attente de transmission.

Tableau 540. Valeurs utilisées pour les zones dans le MQMD distinct (suite)

Zone dans MQMD distinct	Valeur utilisée
<i>PutTime</i>	Heure à laquelle le message a été inséré dans la file d'attente de transmission.
<i>ApplOriginData</i>	Espaces vides
<i>GroupId</i>	MQGI_AUCUN
<i>MsgSeqNumber</i>	1
<i>Offset</i>	0
<i>MsgFlags</i>	MQMF_AUCUN
<i>OriginalLength</i>	MQOL_UNDEFINI

- Sous Windows, la valeur de MQENC\_NATIVE for Micro Focus COBOL diffère de la valeur de C. La valeur de la zone *Encoding* dans le descripteur de message distinct est toujours la valeur de C dans ces environnements ; cette valeur est 546 en décimal. De plus, les zones entières de la structure MQXQH sont dans le codage qui correspond à cette valeur (codage Intel natif).

### Zones du descripteur de message imbriqué

Les zones du descripteur de message imbriqué ont les mêmes valeurs que celles du paramètre **MsgDesc** de l'appel MQPUT ou MQPUT1 , à l'exception de ce qui suit:

- La zone *Version* a toujours la valeur MQMD\_VERSION\_1.
- Si la zone *Priority* a la valeur MQPRI\_PRIORITY\_AS\_Q\_DEF, elle est remplacée par la valeur de l'attribut **DefPriority** de la file d'attente.
- Si la zone *Persistence* a la valeur MQPER\_PERSISTENCE\_AS\_Q\_DEF, elle est remplacée par la valeur de l'attribut **DefPersistence** de la file d'attente.
- Si la zone *MsgId* a la valeur MQMI\_NONE, que l'option MQPMO\_NEW\_MSG\_ID a été spécifiée ou que le message est un message de liste de distribution, *MsgId* est remplacé par un nouvel identificateur de message généré par le gestionnaire de files d'attente.

Lorsqu'un message de liste de distribution est divisé en messages de liste de distribution plus petits placés dans des files d'attente de transmission différentes, la zone *MsgId* de chacun des nouveaux descripteurs de message imbriqués est identique à celle du message de liste de distribution d'origine.

- Si l'option MQPMO\_NEW\_CORREL\_ID a été spécifiée, *CorrelId* est remplacé par un nouvel identificateur de corrélation généré par le gestionnaire de files d'attente.
- Les zones de contexte sont définies comme indiqué par les options MQPMO\_\*\_CONTEXT spécifiées dans le paramètre **PutMsgOpts** ; les zones de contexte sont les suivantes:
  - *AccountingToken*
  - *ApplIdentityData*
  - *ApplOriginData*
  - *PutApplName*
  - *PutApplType*
  - *PutDate*
  - *PutTime*
  - *UserIdentifier*
- Les zones version-2 (si elles étaient présentes) sont supprimées de MQMD et déplacées dans une structure MQMDE, si une ou plusieurs des zones version-2 ont une valeur autre que la valeur par défaut.

## Insertion de messages dans des files d'attente éloignées

Lorsqu'une application insère un message dans une file d'attente éloignée (en spécifiant directement le nom de la file d'attente éloignée ou en utilisant une définition locale de la file d'attente éloignée), le gestionnaire de files d'attente local:

- Crée une structure MQXQH contenant le descripteur de message imbriqué
- Ajoute un MQMDE si nécessaire et qu'il n'est pas déjà présent
- Ajoute les données de message d'application
- Place le message dans une file d'attente de transmission appropriée

## Insertion de messages directement dans les files d'attente de transmission

Une application peut également insérer un message directement dans une file d'attente de transmission. Dans ce cas, l'application doit préfixer les données de message d'application avec une structure MQXQH et initialiser les zones avec les valeurs appropriées. De plus, la zone *Format* du paramètre **MsgDesc** de l'appel MQPUT ou MQPUT1 doit avoir la valeur MQFMT\_XMIT\_Q\_HEADER.

Les données de type caractère de la structure MQXQH créée par l'application doivent être dans le jeu de caractères du gestionnaire de files d'attente local (défini par l'attribut de gestionnaire de files d'attente **CodedCharSetId**) et les données de type entier doivent être dans le codage machine natif. En outre, les données de type caractère dans la structure MQXQH doivent être complétées par des blancs à la longueur définie de la zone ; les données ne doivent pas être arrêtées prématurément à l'aide d'un caractère null, car le gestionnaire de files d'attente ne convertit pas les caractères null et suivants en blancs dans la structure MQXQH.

Toutefois, le gestionnaire de files d'attente ne vérifie pas qu'une structure MQXQH est présente ou que des valeurs valides ont été spécifiées pour les zones.

Les applications ne doivent pas placer leurs messages directement dans SYSTEM.CLUSTER.TRANSMIT.QUEUE.

## Obtention de messages à partir de files d'attente de transmission

Les applications qui extraient des messages d'une file d'attente de transmission doivent traiter les informations de la structure MQXQH de manière appropriée. La présence de la structure MQXQH au début des données de message d'application est indiquée par la valeur MQFMT\_XMIT\_Q\_HEADER renvoyée dans la zone *Format* du paramètre **MsgDesc** de l'appel MQGET. Les valeurs renvoyées dans les zones *CodedCharSetId* et *Encoding* du paramètre **MsgDesc** indiquent le jeu de caractères et le codage des données de type caractère et entier dans la structure MQXQH. Le jeu de caractères et le codage des données de message d'application sont définis par les zones *CodedCharSetId* et *Encoding* dans le descripteur de message imbriqué.

### **StrucId (MQCHAR4) pour MQXQH**

Il s'agit de l'identificateur de structure de la structure d'en-tête de la file d'attente de transmission. Il s'agit toujours d'une zone d'entrée. Sa valeur est MQXQH\_STRUC\_ID.

La valeur doit être:

#### **ID\_STRUC\_MQXQH\_**

Identificateur de la structure d'en-tête de file d'attente de transmission.

Pour le langage de programmation C, la constante MQXQH\_STRUC\_ID\_ARRAY est également définie. Cette valeur est identique à MQXQH\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### **Version (MQLONG) pour MQXQH**

Il s'agit du numéro de version de la structure. La valeur doit être:

## **MQXQH\_VERSION\_1**

Numéro de version de la structure d'en-tête de file d'attente de transmission.

La constante suivante indique le numéro de version de la version en cours:

## **MQXQH\_CURRENT\_VERSION**

Version actuelle de la structure d'en-tête de file d'attente de transmission.

La valeur initiale de cette zone est MQXQH\_VERSION\_1.

## **RemoteQName (MQCHAR48) pour MQXQH**

Il s'agit du nom de la file d'attente de messages correspondant à la destination finale apparente du message (cette dernière peut ne pas être la destination finale si, par exemple, cette file d'attente est définie dans *RemoteQMgrName* comme étant une définition locale d'une autre file d'attente éloignée).

Si le message est un message de liste de distribution (c'est-à-dire que la zone *Format* du descripteur de message imbriqué est MQFMT\_DIST\_HEADER), *RemoteQName* est vide.

La longueur de cette zone est indiquée par MQ\_Q\_NAME\_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

## **RemoteQMgrNom (MQCHAR48) pour MQXQH**

Il s'agit du nom du gestionnaire de files d'attente ou du groupe de partage de files d'attente propriétaire de la file d'attente qui est la destination finale apparente du message.

Si le message est un message de liste de distribution, *RemoteQMgrName* est vide.

La longueur de cette zone est indiquée par MQ\_Q\_MGR\_NAME\_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

## **MsgDesc (MQMD1) pour MQXQH**

Il s'agit du descripteur de message intégré. Il s'agit d'une copie fermée du descripteur de message MQMD qui a été spécifié en tant que paramètre **MsgDesc** dans l'appel MQPUT ou MQPUT1 lors de l'insertion initiale du message dans la file d'attente éloignée.


**Remarque :** Il s'agit d'un MQMD version-1 .

Les valeurs initiales des zones de cette structure sont identiques à celles de la structure MQMD.

## **Appels de fonctions**

Cette section fournit des informations sur tous les appels MQI possibles. Les descriptions, la syntaxe, les informations de paramètre, les remarques d'utilisation et les appels de langue pour chaque langue possible sont indiqués pour chacun des différents appels.

### **Référence associée**

 [Exemples de sortie CEDF des appels MQI](#)

## **Descriptions des appels**

Cette section décrit les appels MQI.

- [«MQBACK-Annulation des modifications», à la page 657](#)
- [«MQBEGIN-Début de l'unité de travail», à la page 661](#)
- [«MQBUFMH-Conversion de la mémoire tampon en descripteur de message», à la page 664](#)
- [«MQCB-Gestion des rappels», à la page 668](#)
- [«MQCB\\_FUNCTION-Fonction de rappel», à la page 678](#)
- [«MQCLOSE-Fermer l'objet», à la page 679](#)
- [«MQCMIT-Validation des modifications», à la page 688](#)
- [«MQCONN-Connexion du gestionnaire de files d'attente», à la page 692](#)

- «MQCONN - Connexion du gestionnaire de files d'attente (étendue)», à la page 700
- «MQCRTMH-Création d'un descripteur de message», à la page 705
- «MQCTL-Rappels de contrôle», à la page 709
- «MQDISC-Déconnexion du gestionnaire de files d'attente», à la page 715
- «MQDLTMH-Suppression du descripteur de message», à la page 719
- «MQDLTMP-Propriété de message Delete», à la page 722
- «Message d'extraction MQGET», à la page 724
- «MQINQ-Attributs d'objet Inquire», à la page 737
- «MQINQMP-Propriété de message d'interrogation», à la page 757
- «MQMHBUFF-Conversion du descripteur de message en mémoire tampon», à la page 762
- «MQOPEN-Objet ouvert», à la page 766
- «MQPUT-Insertion d'un message», à la page 785
- «MQPUT1 -Message d'insertion unique», à la page 799
- «MQSET-Définition des attributs d'objet», à la page 810
- «MQSETMP-Définition de la propriété de message», à la page 816
- «MQSTAT-Extraction des informations de statut», à la page 821
- «MQMHBUFF-Conversion du descripteur de message en mémoire tampon», à la page 762
- «MQSUB-Enregistrement d'abonnement», à la page 825
- «MQSUBRQ-Demande d'abonnement», à la page 832

L'aide en ligne sur les plateformes UNIX , sous la forme de pages *man* , est disponible pour ces appels.

**Remarque :** Les appels associés à la conversion de données, MQXCNV et MQ\_DATA\_CONV\_EXIT, se trouvent dans «Exit de conversion de données», à la page 951.

### ***Conventions utilisées dans les descriptions d'appel***

Pour chaque appel, cette collection de rubriques fournit une description des paramètres et de l'utilisation de l'appel dans un format indépendant du langage de programmation. Elle est suivie d'appels typiques de l'appel et de déclarations typiques de ses paramètres, dans chacun des langages de programmation pris en charge.

**Important :** Lorsque vous codez des appels d'API IBM MQ , vous devez vous assurer que tous les paramètres appropriés (comme décrit dans les sections suivantes) sont fournis. Si vous ne le faites pas, vous risquez d'obtenir des résultats imprévisibles.

La description de chaque appel contient les sections suivantes:

#### **Nom de l'appel**

Nom de l'appel, suivi d'une brève description de l'objet de l'appel.

#### **Paramètres**

Pour chaque paramètre, le nom est suivi de son type de données entre parenthèses () et l'un des éléments suivants:

##### **entrée**

Vous fournissez des informations dans le paramètre lorsque vous effectuez l'appel.

##### **sortie**

Le gestionnaire de files d'attente renvoie des informations dans le paramètre lorsque l'appel aboutit ou échoue.

##### **d'entrée-sortie**

Vous fournissez des informations dans le paramètre lorsque vous effectuez l'appel et le gestionnaire de files d'attente modifie les informations lorsque l'appel se termine ou échoue.

Exemple :

## Compcode (MQLONG)-sortie

Dans certains cas, le type de données est une structure. Dans tous les cas, vous trouverez plus d'informations sur le type de données ou la structure dans [«Types de données élémentaires»](#), à la page 238.

Les deux derniers paramètres de chaque appel sont un code achèvement et un code anomalie. Le code achèvement indique si l'appel a abouti, partiellement ou pas du tout. Des informations supplémentaires sur la réussite partielle ou l'échec de l'appel sont fournies dans le code anomalie. Pour plus d'informations sur chaque code achèvement et anomalie, voir [«Codes retour»](#), à la page 917.

### Remarques sur l'utilisation

Informations supplémentaires sur l'appel, décrivant la façon de l'utiliser et les éventuelles restrictions liées à son utilisation.

### Appel du langage assembleur

Appel standard de l'appel et déclaration de ses paramètres, en langage assembleur.

### Appel C

Appel standard de l'appel et déclaration de ses paramètres, en C.

### Appel COBOL

Appel standard de l'appel et déclaration de ses paramètres en COBOL.

### Appel PL/I

Appel typique de l'appel, et déclaration de ses paramètres, en PL/I.

Tous les paramètres sont transmis par référence.

### Appel Visual Basic

Appel standard de l'appel et déclaration de ses paramètres dans Visual Basic.

Les autres conventions de notation sont les suivantes:

### Constantes

Les noms des constantes sont affichés en majuscules ; par exemple, MQOO\_OUTPUT. Un ensemble de constantes ayant le même préfixe est présenté comme suit: MQIA\_\*. Pour la valeur d'une constante, voir [«Constantes»](#), à la page 61 .

### Tableaux

Dans certains appels, les paramètres sont des tableaux de chaînes de caractères qui n'ont pas de taille fixe. Dans les descriptions de ces paramètres, une minuscule n représente une constante numérique. Lorsque vous codez la déclaration pour ce paramètre, remplacez le n par la valeur numérique requise.

## Utilisation des appels en langage C

Les paramètres qui sont *uniquement en entrée* et de type MQHCONN, MQHOBJ, MQHMSG ou MQLONG sont transmis par valeur. Pour tous les autres paramètres, l' *adresse* du paramètre est transmise par valeur.

Il n'est pas nécessaire de spécifier tous les paramètres transmis par adresse chaque fois que vous appelez une fonction. Si vous n'avez pas besoin d'un paramètre particulier, spécifiez un pointeur null comme paramètre sur l'appel de fonction, à la place de l'adresse des données de paramètre. Les paramètres pour lesquels cela est possible sont identifiés dans les descriptions d'appel.

Aucun paramètre n'est renvoyé comme valeur de l'appel ; dans la terminologie C, cela signifie que tous les appels renvoient void.

### Déclaration du paramètre Buffer

Les appels **MQGET**, **MQPUT** et **MQPUT1** comportent chacun un paramètre dont le type de données n'est pas défini: le paramètre *Buffer* . Utilisez ce paramètre pour envoyer et recevoir les données de message de l'application.



Les paramètres de ce type sont présentés dans les exemples C sous la forme de tableaux de MQBYTE. Vous pouvez déclarer les paramètres de cette manière, mais il est généralement plus pratique de les déclarer comme la structure particulière qui décrit la présentation des données dans le message. Le prototype de fonction déclare le paramètre en tant que pointeur vers vide, de sorte que vous pouvez spécifier l'adresse de n'importe quel type de données en tant que paramètre sur l'appel d'appel.

Pointer-to-void est un pointeur vers des données de format non défini. Il est défini comme suit:

```
typedef void *PMQVOID;
```

## MQBACK-Annulation des modifications

L'appel MQBACK indique au gestionnaire de files d'attente que tous les extractions et insertions de messages qui se sont produites depuis le dernier point de synchronisation doivent être annulées.

Les messages insérés dans une unité d'oeuvre sont supprimés ; les messages extraits dans une unité d'oeuvre sont réintégrés dans la file d'attente.

- Sous z/OS, cet appel est utilisé uniquement par les programmes de traitement par lots (y compris les programmes DL/I de traitement par lots IMS).

## Syntaxe

MQBACK (*Hconn*, *Code achèvement*, *Raison*)

## Paramètres

### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

### CompCode

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

#### MQCC\_OK

Achèvement réussi.

#### MQCC\_WARNING

Avertissement (achèvement partiel).

#### MQCC\_FAILED

Echec de l'appel.

### Motif

Type : MQLONG - sortie

Si *CompCode* est MQCC\_OK:

#### MQRC\_NONE

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_WARNING:

#### MQRC\_OUTCOME\_EN attente

(2124, X'84C') Le résultat de l'opération d'inexécution est en attente.

Si *CompCode* est MQCC\_FAILED:

#### MQRC\_ADAPTER\_SERV\_LOAD\_ERROR

(2130, X'852') Impossible de charger le module de service d'adaptateur.

**MQRC\_API\_EXIT\_ERROR**

(2374, X'946') Echec de l'exit API.

**MQRC\_ASID\_MISMATCH**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

**MQRC\_CF\_STRUC\_IN\_USE**

(2346, X'92A') Structure de l'unité de couplage en cours d'utilisation.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**MQRC\_ENVIRONMENT\_ERROR**

(2012, X'7DC') Appel non valide dans l'environnement.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') Descripteur de connexion non valide.

**MQRC\_OBJECT\_DAMAGED**

(2101, X'835') Objet endommagé.

**MQRC\_OUTCOME\_MIXTE**

(2123, X'84B') Le résultat de l'opération de validation ou d'exclusion est mélangé.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') Ressources système disponibles insuffisantes.

**MQRC\_STORAGE\_MEDIUM\_FULL**

(2192, X'890') Support de stockage externe saturé.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') Mémoire disponible insuffisante.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#)

## Remarques sur l'utilisation

1. Vous pouvez utiliser cet appel uniquement lorsque le gestionnaire de files d'attente lui-même coordonne l'unité d'oeuvre. Ce nom peut être :
  - Unité d'oeuvre locale dans laquelle les modifications affectent uniquement les ressources MQ .
  - Unité de travail globale dans laquelle les modifications peuvent affecter les ressources appartenant à d'autres gestionnaires de ressources, ainsi que les ressources MQ .

Pour plus de détails sur les unités d'oeuvre locales et globales, voir [«MQBEGIN-Début de l'unité de travail»](#), à la page 661.
2. Dans les environnements où le gestionnaire de files d'attente ne coordonne pas l'unité d'oeuvre, utilisez l'appel d'annulation approprié à la place de MQBACK. L'environnement peut également prendre en charge une interruption implicite provoquée par l'arrêt anormal de l'application.
  - Sous z/OS, utilisez les appels suivants:
    - Les programmes batch (y compris les programmes IMS batch DL/I) peuvent utiliser l'appel MQBACK si l'unité de travail n'affecte que les ressources MQ . Toutefois, si l'unité de travail affecte à la fois les ressources MQ et les ressources appartenant à d'autres gestionnaires de ressources (par exemple, Db2 ), utilisez l'appel SRRBACK fourni par le service RRS ( z/OS Recoverable Resource Service). L'appel SRRBACK annule les modifications apportées aux ressources appartenant aux gestionnaires de ressources qui ont été activés pour la coordination RRS.

- Les applications CICS doivent utiliser la commande EXEC CICS SYNCPOINT ROLLBACK pour rétablir l'unité d'oeuvre. N'utilisez pas l'appel MQBACK pour les applications CICS .
  - Les applications IMS (autres que les programmes batch DL/I) doivent utiliser des appels IMS tels que ROLB pour rétablir l'unité d'oeuvre. N'utilisez pas l'appel MQBACK pour les applications IMS (autres que les programmes batch DL/I).
- Sous IBM i, utilisez cet appel pour les unités de travail locales coordonnées par le gestionnaire de files d'attente. Cela signifie qu'une définition de validation ne doit pas exister au niveau du travail, c'est-à-dire que la commande STRCMTCTL avec le paramètre **CMTSCOPE(\*JOB)** ne doit pas avoir été émise pour le travail.
3. Si une application se termine avec des modifications non validées dans une unité d'oeuvre, la disposition de ces modifications varie selon que l'application se termine normalement ou anormalement. Pour plus de détails, voir les remarques sur l'utilisation dans «MQDISC-Déconnexion du gestionnaire de files d'attente», à la page 715 .
  4. Lorsqu'une application insère ou extrait des messages dans des groupes ou des segments de messages logiques, le gestionnaire de files d'attente conserve les informations relatives au groupe de messages et au message logique pour les derniers appels MQPUT et MQGET ayant abouti. Ces informations sont associées à l'identificateur de file d'attente et incluent les éléments suivants:
    - Valeurs des zones *GroupId*, *MsgSeqNumber*, *Offset* et *MsgFlags* dans MQMD.
    - Indique si le message fait partie d'une unité de travail.
    - Pour l'appel MQPUT: indique si le message est persistant ou non persistant.

Le gestionnaire de files d'attente conserve *trois* ensembles d'informations de groupe et de segment, un ensemble pour chacun des éléments suivants:

    - Dernier appel MQPUT réussi (qui peut faire partie d'une unité de travail).
    - Dernier appel MQGET réussi qui a supprimé un message de la file d'attente (il peut faire partie d'une unité d'oeuvre).
    - Dernier appel MQGET ayant réussi à parcourir un message dans la file d'attente (il ne peut pas faire partie d'une unité d'oeuvre).
  5. Les informations associées à l'appel MQGET sont restaurées à la valeur qu'elles avaient avant le premier appel MQGET réussi pour ce descripteur de file d'attente dans l'unité d'oeuvre en cours.
 

Les files d'attente qui ont été mises à jour par l'application après le démarrage de l'unité de travail, mais en dehors de la portée de l'unité de travail, ne voient pas leurs informations de groupe et de segment restaurées si l'unité de travail est annulée.

La restauration des informations de groupe et de segment à sa valeur précédente lors de l'annulation d'une unité de travail permet à l'application de répartir un grand groupe de messages ou un grand message logique composé de plusieurs segments sur plusieurs unités de travail, et de redémarrer au point approprié du groupe de messages ou du message logique si l'une des unités de travail échoue.

L'utilisation de plusieurs unités d'oeuvre peut s'avérer avantageuse si le gestionnaire de files d'attente local ne dispose que d'une mémoire de file d'attente limitée. Toutefois, l'application doit conserver suffisamment d'informations pour pouvoir redémarrer l'insertion ou l'obtention de messages au point approprié en cas de défaillance du système.

Pour plus d'informations sur le redémarrage au point approprié après une défaillance du système, voir l'option MQPMO\_LOGICAL\_ORDER décrite dans «MQPMO-Options de message d'insertion», à la page 522 et l'option MQGMO\_LOGICAL\_ORDER décrite dans «Options MQGMO-Get-message», à la page 382.

Les autres remarques d'utilisation s'appliquent uniquement lorsque le gestionnaire de files d'attente coordonne les unités de travail.
  6. Une unité de travail a la même portée qu'un descripteur de connexion. Tous les appels MQ qui affectent une unité de travail particulière doivent être exécutés à l'aide du même descripteur de connexion. Les appels émis à l'aide d'un descripteur de connexion différent (par exemple, les appels émis par une autre application) affectent une unité d'oeuvre différente. Pour plus d'informations sur la

portée des descripteurs de connexion, voir le paramètre **Hconn** décrit dans «MQCONN-Connexion du gestionnaire de files d'attente», à la page 692 .

7. Seuls les messages insérés ou extraits dans le cadre de l'unité d'oeuvre en cours sont affectés par cet appel.
8. Une application à exécution longue qui émet des appels MQGET, MQPUT ou MQPUT1 dans une unité de travail, mais qui n'émet jamais d'appel de validation ou d'annulation, peut remplir les files d'attente avec des messages qui ne sont pas disponibles pour d'autres applications. Pour éviter cette possibilité, l'administrateur doit définir l'attribut de gestionnaire de files d'attente **MaxUncommittedMsgs** sur une valeur suffisamment faible pour empêcher les applications en boucle indéfinie de remplir les files d'attente, mais suffisamment élevée pour permettre aux applications de messagerie attendues de fonctionner correctement.

## Appel C

```
MQBACK (Hconn, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;      /* Connection handle */
MQLONG   CompCode;   /* Completion code */
MQLONG   Reason;     /* Reason code qualifying CompCode */
```

## Appel COBOL

```
CALL 'MQBACK' USING HCONN, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle
01 HCONN      PIC S9(9) BINARY.
** Completion code
01 COMPCODE   PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON     PIC S9(9) BINARY.
```

## Appel PL/I

```
call MQBACK (Hconn, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
dcl Hconn      fixed bin(31); /* Connection handle */
dcl CompCode   fixed bin(31); /* Completion code */
dcl Reason     fixed bin(31); /* Reason code qualifying CompCode */
```

## Appel d'assembleur de haut niveau

```
CALL MQBACK, (HCONN, COMPCODE, REASON)
```

Déclarez les paramètres comme suit :

```
HCONN      DS F Connection handle
```

```
COMPCODE DS F Completion code
REASON   DS F Reason code qualifying COMPCODE
```

## Appel Visual Basic

```
MQBACK Hconn, CompCode, Reason
```

Déclarez les paramètres comme suit :

```
Dim Hconn      As Long 'Connection handle'
Dim CompCode   As Long 'Completion code'
Dim Reason     As Long 'Reason code qualifying CompCode'
```

## MQBEGIN-Début de l'unité de travail

L'appel MQBEGIN démarre une unité de travail coordonnée par le gestionnaire de files d'attente et qui peut impliquer des gestionnaires de ressources externes.

### Syntaxe

MQBEGIN (*Hconn*, *BeginOptions*, *Code achèvement*, *Motif*)

### Paramètres

#### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

*Hconn* doit être un descripteur de connexion non partagé. Si un descripteur de connexion partagée est spécifié, l'appel échoue avec le code anomalie MQRC\_HCONN\_ERROR. Pour plus d'informations sur les descripteurs partagés et non partagés, voir la description des options MQCNO\_HANDLE\_SHARE\_\* dans [«MQCNO-Options de connexion»](#), à la page 325 .

#### BeginOptions

Type: MQBO-entrée / sortie

Il s'agit d'options qui contrôlent l'action de MQBEGIN, comme décrit dans [«MQBO-Options de début»](#), à la page 286.

Si aucune option n'est requise, les programmes écrits en assembleur C ou S/390 peuvent spécifier une adresse de paramètre null au lieu de spécifier l'adresse d'une structure MQBO.

#### CompCode

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

##### MQCC\_OK

Achèvement réussi.

##### MQCC\_WARNING

Avertissement (achèvement partiel).

##### MQCC\_FAILED

Echec de l'appel.

#### Motif

Type : MQLONG - sortie

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_WARNING:

**MQRC\_NO\_EXTERNAL\_PARTICIPANTS**

(2121, X'849') Aucun gestionnaire de ressources participant enregistré.

**MQRC\_PARTICIPANT\_NOT\_XX\_ENCODE\_CASE\_ONE disponible**

(2122, X'84A') Gestionnaire de ressources participant non disponible.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_API\_EXIT\_ERROR**

(2374, X'946') Echec de l'exit API.

**MQRC\_BO\_ERREUR**

(2134, X'856') Structure des options de début incorrecte.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**MQRC\_ENVIRONMENT\_ERROR**

(2012, X'7DC') Appel non valide dans l'environnement.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') Descripteur de connexion non valide.

**MQRC\_OPTIONS\_ERROR**

(2046, X'7FE') Options incorrectes ou incohérentes.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') Ressources système disponibles insuffisantes.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') Mémoire disponible insuffisante.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') Une erreur inattendue s'est produite.

**MQRC\_UOW\_EN\_COURS**

(2128, X'850') Unité de travail déjà démarrée.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Remarques sur l'utilisation

1. Utilisez l'appel MQBEGIN pour démarrer une unité de travail coordonnée par le gestionnaire de files d'attente et qui peut impliquer des modifications apportées aux ressources appartenant à d'autres gestionnaires de ressources. Le gestionnaire de files d'attente prend en charge trois types d'unité de travail:

- **Unité de travail locale coordonnée par le gestionnaire de files d'attente:** unité de travail dans laquelle le gestionnaire de files d'attente est le seul gestionnaire de ressources participant, de sorte que le gestionnaire de files d'attente joue le rôle de coordinateur d'unité de travail.
  - Pour démarrer ce type d'unité de travail, spécifiez l'option MQPMO\_SYNCPOINT ou MQGMO\_SYNCPOINT sur le premier appel MQPUT, MQPUT1 ou MQGET dans l'unité de travail.
  - Pour valider ou rétablir ce type d'unité de travail, utilisez l'appel MQCMIT ou MQBACK.
- **Unité de travail globale coordonnée par le gestionnaire de files d'attente:** unité de travail dans laquelle le gestionnaire de files d'attente agit en tant que coordinateur d'unité de travail, à la fois pour les ressources MQ et pour les ressources appartenant à d'autres gestionnaires de ressources.

Ces gestionnaires de ressources coopèrent avec le gestionnaire de files d'attente pour s'assurer que toutes les modifications apportées aux ressources de l'unité d'oeuvre sont validées ou annulées ensemble.

– Pour démarrer ce type d'unité de travail, utilisez l'appel MQBEGIN.

– Pour valider ou rétablir ce type d'unité d'oeuvre, utilisez les appels MQCMIT et MQBACK.

- **Unité de travail globale coordonnée en externe:** unité de travail dans laquelle le gestionnaire de files d'attente est un participant, mais le gestionnaire de files d'attente n'agit pas en tant que coordinateur d'unité de travail. A la place, il existe un coordinateur d'unité de travail externe avec lequel le gestionnaire de files d'attente coopère.

– Pour démarrer ce type d'unité de travail, utilisez l'appel approprié fourni par le coordinateur de l'unité de travail externe.

Si l'appel MQBEGIN est utilisé pour tenter de démarrer l'unité de travail, l'appel échoue avec le code anomalie MQRC\_ENVIRONMENT\_ERROR.

– Pour valider ou exécuter ce type d'unité de travail, utilisez les appels de validation et d'exécution fournis par le coordinateur d'unité de travail externe.

Si vous utilisez l'appel MQCMIT ou MQBACK pour valider ou rétablir l'unité d'oeuvre, l'appel échoue avec le code anomalie MQRC\_ENVIRONMENT\_ERROR.

2. Si l'application se termine avec des modifications non validées dans une unité de travail, la disposition de ces modifications varie selon que l'application se termine normalement ou anormalement. Pour plus de détails, voir les remarques sur l'utilisation dans [«MQDISC-Déconnexion du gestionnaire de files d'attente»](#), à la page 715 .
3. Une application ne peut participer qu'à une seule unité de travail à la fois. L'appel MQBEGIN échoue avec le code anomalie MQRC\_UOW\_IN\_PROGRESS s'il existe déjà une unité de travail pour l'application, quel que soit le type d'unité de travail.
4. L'appel MQBEGIN n'est pas valide dans un environnement client MQ MQI. Une tentative d'utilisation de l'appel échoue avec le code anomalie MQRC\_ENVIRONMENT\_ERROR.
5. Lorsque le gestionnaire de files d'attente joue le rôle de coordinateur d'unité de travail pour les unités de travail globales, les gestionnaires de ressources pouvant participer à l'unité de travail sont définis dans le fichier de configuration du gestionnaire de files d'attente.
6. Sous IBM i, les trois types d'unité de travail sont pris en charge comme suit:
  - **L'unité d'oeuvre locale coordonnée par le gestionnaire de files d'attente** ne peut être utilisée que lorsqu'une définition de validation n'existe pas au niveau du travail, c'est-à-dire que la commande STRCMTCTL avec le paramètre **CMTSCOPE(\*JOB)** ne doit pas avoir été émise pour le travail.
  - **L'unité d'oeuvre globale coordonnée par le gestionnaire de files d'attente** n'est pas prise en charge.
  - **L'unité d'oeuvre globale coordonnée en externe** ne peut être utilisée que lorsqu'une définition de validation existe au niveau du travail, c'est-à-dire que la commande STRCMTCTL avec le paramètre **CMTSCOPE(\*JOB)** doit avoir été émise pour le travail. Si tel est le cas, les opérations IBM i COMMIT et ROLLBACK s'appliquent aux ressources MQ ainsi qu'aux ressources appartenant à d'autres gestionnaires de ressources participants.

## Appel C

```
MQBEGIN (Hconn, &BeginOptions, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;           /* Connection handle */
MQBO     BeginOptions;   /* Options that control the action of MQBEGIN */
MQLONG   CompCode;      /* Completion code */
MQLONG   Reason;        /* Reason code qualifying CompCode */
```

## Appel COBOL

```
CALL 'MQBEGIN' USING HCONN, BEGINOPTIONS, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle
01 HCONN          PIC S9(9) BINARY.
** Options that control the action of MQBEGIN
01 BEGINOPTIONS.
   COPY CMQBOV.
** Completion code
01 COMPCODE       PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON         PIC S9(9) BINARY.
```

## Appel PL/I

```
call MQBEGIN (Hconn, BeginOptions, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
dcl Hconn          fixed bin(31); /* Connection handle */
dcl BeginOptions  like MQBO;     /* Options that control the action of
                                MQBEGIN */
dcl CompCode      fixed bin(31); /* Completion code */
dcl Reason        fixed bin(31); /* Reason code qualifying CompCode */
```

## Appel Visual Basic

```
MQBEGIN Hconn, BeginOptions, CompCode, Reason
```

Déclarez les paramètres comme suit :

```
Dim Hconn          As Long 'Connection handle'
Dim BeginOptions  As MQBO 'Options that control the action of MQBEGIN'
Dim CompCode      As Long 'Completion code'
Dim Reason        As Long 'Reason code qualifying CompCode'
```

## MQBUFMH-Conversion de la mémoire tampon en descripteur de message

L'appel de fonction MQBUFMH convertit une mémoire tampon en descripteur de message et est l'inverse de l'appel MQMHBUF.

Cet appel prend un descripteur de message et des propriétés MQRFH2 dans la mémoire tampon et les rend disponibles via un descripteur de message. Les propriétés MQRFH2 des données de message sont éventuellement supprimées. Les zones *Encoding*, *CodedCharSetIdet Format* du descripteur de message sont mises à jour, si nécessaire, pour décrire correctement le contenu de la mémoire tampon après la suppression des propriétés.

### Syntaxe

```
MQBUFMH (Hconn, Hmsg, BufMsgHOpts, MsgDesc, BufferLength, Buffer, DataLength, Compcode, Motif)
```

### Paramètres

#### Hconn

Type : MQHCONN - entrée



Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de **Hconn** doit correspondre au descripteur de connexion qui a été utilisé pour créer le descripteur de message spécifié dans le paramètre **Hmsg**.

Si le descripteur de message a été créé à l'aide de MQHC\_UNASSOCIATED\_HCONN, une connexion valide doit être établie sur l'unité d'exécution qui convertit une mémoire tampon en descripteur de message. Si aucune connexion valide n'est établie, l'appel échoue avec MQRC\_CONNECTION\_BROKEN.

### **Msg**

Type: MQHMQSG-entrée

Il s'agit du descripteur de message pour lequel une mémoire tampon est requise. La valeur a été renvoyée par un appel MQCRTMH précédent.

### **BufMsgHOpts**

Type: MQBMHO-entrée

La structure MQBMHO permet aux applications de spécifier des options qui contrôlent la façon dont les descripteurs de message sont produits à partir des mémoires tampon.

Pour plus d'informations, voir «MQBMHO-Options de traitement de la mémoire tampon vers le message», à la page 282.

### **MsgDesc**

Type : MQMD - entrée/sortie

La structure *MsgDesc* contient les propriétés de descripteur de message et décrit le contenu de la zone tampon.

En sortie de l'appel, les propriétés sont éventuellement supprimées de la zone tampon et, dans ce cas, le descripteur de message est mis à jour pour décrire correctement la zone tampon.

Les données de cette structure doivent être dans le jeu de caractères et le codage de l'application.

### **BufferLength**

Type : MQLONG - entrée

*BufferLength* est la longueur de la zone tampon, en octets.

Une valeur *BufferLength* de zéro octet est valide et indique que la zone tampon ne contient aucune donnée.

### **Mémoire tampon**

Type: MQBYTEExBufferLength-input/output

Il s'agit d'options qui contrôlent l'action de MQBEGIN, comme décrit dans «MQBEGIN-Début de l'unité de travail», à la page 661.

**Buffer** définit la zone contenant la mémoire tampon de messages. Pour la plupart des données, vous devez aligner la mémoire tampon sur une limite de 4 octets.

Si **Buffer** contient des données de type caractère ou numérique, définissez les zones *CodedCharSetId* et *Encoding* du paramètre **MsgDesc** sur les valeurs appropriées aux données ; cela permet la conversion des données, si nécessaire.

Si des propriétés sont trouvées dans la mémoire tampon de messages, elles sont éventuellement supprimées ; elles sont ensuite disponibles à partir du descripteur de message lors du retour de l'appel.

Dans le langage de programmation C, le paramètre est déclaré comme un pointeur vers vide, ce qui signifie que l'adresse de tout type de données peut être spécifiée comme paramètre.

Si la valeur du paramètre **BufferLength** est zéro, il n'est pas fait référence à **Buffer** ; dans ce cas, l'adresse de paramètre transmise par les programmes écrits en assembleur C ou System/390 peut être null.

**DataLength**

Type : MQLONG - sortie

Longueur, en octets, de la mémoire tampon dont les propriétés peuvent être supprimées.

**CompCode**

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

**MQCC\_OK**

Achèvement réussi.

**MQCC\_FAILED**

Echec de l'appel.

**Motif**

Type : MQLONG - sortie

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'089C') Carte non disponible.

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

**MQRC\_ASID\_MISMATCH**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

**MQRC\_BMHO\_ERREUR**

(2489, X'09B9') La structure des options de la mémoire tampon vers le descripteur de message n'est pas valide.

**MQRC\_BUFFER\_ERROR**

(2004, X'07D4') Paramètre de mémoire tampon non valide.

**MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'07D5') Paramètre de longueur de mémoire tampon non valide.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'07D9') La connexion au gestionnaire de files d'attente a été perdue.

**ERREUR MQRC\_HMSG\_ERROR**

(2460, X'099C') Descripteur de message non valide.

**MQRC\_MD\_ERROR**

(2026, X'07EA') Descripteur de message non valide.

**MQRC\_MSG\_HANDLE\_IN\_USE**

(2499, X'09C3') Descripteur de message déjà utilisé.

**MQRC\_OPTIONS\_ERROR**

(2046, X'07FE') Options non valides ou non cohérentes.

**MQRC\_RFH\_ERREUR**

(2334, X'091E') Structure MQRFH2 non valide.

**MQRC\_RFH\_FORMAT\_ERREUR**

(2421, X'0975') Un dossier MQRFH2 contenant des propriétés n'a pas pu être analysé.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Remarques sur l'utilisation

Les appels MQBUFMH ne peuvent pas être interceptés par des exits API-une mémoire tampon est convertie en descripteur de message dans l'espace d'application ; l'appel n'atteint pas le gestionnaire de files d'attente.

## Appel C

```
MQBUFMH (Hconn, Hmsg, &BufMsgHOpts, &MsgDesc, BufferLength, Buffer,
         &DataLength, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN Hconn;          /* Connection handle */
MQHMSG  Hmsg;           /* Message handle */
MQBMHO  BufMsgHOpts;   /* Options that control the action of MQBUFMH */
MQMD    MsgDesc;       /* Message descriptor */
MQLONG  BufferLength;   /* Length in bytes of the Buffer area */
MQBYTE  Buffer[n];      /* Area to contain the message buffer */
MQLONG  DataLength;    /* Length of the output buffer */
MQLONG  CompCode;      /* Completion code */
MQLONG  Reason;        /* Reason code qualifying CompCode */
```

## Appel COBOL

```
CALL 'MQBUFMH' USING HCONN, HMSG, BUFMSGHOPTS, MSGDESC, BUFFERLENGTH,
                   BUFFER, DATALENGTH, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle
01 HCONN          PIC S9(9) BINARY.
** Message handle
01 HMSG           PIC S9(18) BINARY.
** Options that control the action of MQBUFMH
01 BUFMSGHOPTS.
   COPY CMQBMHOV.
** Message descriptor
01 MSGDESC.
   COPY CMQMD.
** Length in bytes of the Buffer area
01 BUFFERLENGTH PIC S9(9) BINARY.
** Area to contain the message buffer
01 BUFFER        PIC X(n).
** Length of the output buffer
01 DATALENGTH  PIC S9(9) BINARY.
** Completion code
01 COMPCODE     PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON       PIC S9(9) BINARY.
```

## Appel PL/I

```
call MQBUFMH (Hconn, Hmsg, BufMsgHOpts, MsgDesc, BufferLength, Buffer,
             DataLength, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
dcl Hconn          fixed bin(31); /* Connection handle */
dcl Hmsg           fixed bin(63); /* Message handle */
dcl BufMsgHOpts    like MQBMHO;   /* Options that control the action of
                                   MQBUFMH */
dcl MsgDesc        like MQMD;     /* Message descriptor */
dcl BufferLength    fixed bin(31); /* Length in bytes of the Buffer area */
```

```

dcl Buffer          char(n);          /* Area to contain the message buffer */
dcl DataLength    fixed bin(31);    /* Length of the output buffer */
dcl CompCode      fixed bin(31);    /* Completion code */
dcl Reason        fixed bin(31);    /* Reason code qualifying CompCode */

```

## Appel d'assembleur de haut niveau

```

CALL MQBUFMH, (HCONN, HMSG, BUFMSGHOPTS, MSGDESC, BUFFERLENGTH, BUFFER,
              DATALENGTH, COMPCODE, REASON)

```

Déclarez les paramètres comme suit :

HCONN	DS	F	Connection handle
HMSG	DS	D	Message handle
BUFMSGHOPTS	CMQBMHOA	,	Options that control the action of MQBUFMH
MSGDESC	CMQMDA	,	Message descriptor
BUFFERLENGTH	DS	F	Length in bytes of the BUFFER area
BUFFER	DS	CL(n)	Area to contain the properties
DATALENGTH	DS	F	Length of the output buffer
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## MQCB-Gestion des rappels

L'appel MQCB enregistre un rappel pour le descripteur d'objet spécifié et contrôle l'activation et les modifications apportées au rappel.

Un rappel est un élément de code (spécifié comme le nom d'une fonction qui peut être liée dynamiquement ou comme pointeur de fonction) qui est appelé par IBM MQ lorsque certains événements se produisent.

Pour utiliser MQCB et MQCTL sur un client, vous devez être connecté à un serveur sur lequel le paramètre **SHARECNV** négocié du canal a accepté une valeur différente de zéro.

Les types de rappel pouvant être définis sont les suivants:

### Consommateur de message

Une fonction de rappel de consommateur de message est appelée lorsqu'un message répondant aux critères de sélection spécifiés est disponible sur un descripteur d'objet.

Une seule fonction de rappel peut être enregistrée pour chaque descripteur d'objet. Si une seule file d'attente doit être lue avec plusieurs critères de sélection, la file d'attente doit être ouverte plusieurs fois et une fonction de consommateur doit être enregistrée sur chaque descripteur.

### Gestionnaire d'événements

Le gestionnaire d'événements est appelé pour les conditions qui affectent l'ensemble de l'environnement de rappel.

La fonction est appelée lorsqu'une condition d'événement se produit, par exemple, l'arrêt ou la mise au repos d'un gestionnaire de files d'attente ou d'une connexion.

La fonction n'est pas appelée pour des conditions spécifiques à un consommateur de message unique, par exemple MQRC\_GET\_INHIBITED; elle est toutefois appelée si une fonction de rappel ne se termine pas normalement.

## Syntaxe

MQCB (*Hconn, Opération, CallbackDesc, Hobj, MsgDesc, GetMsgOpts, CompCode, Motif*)

## Paramètres

### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Dans les applications z/OS for CICS , vous pouvez spécifier la valeur spéciale suivante pour *MQHC\_DEF\_HCONN* afin d'utiliser le descripteur de connexion associé à cette unité d'exécution.

### Opération

Type : MQLONG - entrée

Opération en cours de traitement sur le rappel défini pour le descripteur d'objet spécifié. Vous devez spécifier l'une des options suivantes. Pour spécifier plusieurs options, ajoutez les valeurs ensemble (n'ajoutez pas la même constante plusieurs fois) ou combinez les valeurs à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations bit).

#### **MQOP\_REGISTER**

Définissez la fonction de rappel pour le descripteur d'objet spécifié. Cette opération définit la fonction à appeler et les critères de sélection à utiliser.

Si une fonction de rappel est déjà définie pour le descripteur d'objet, la définition est remplacée. Si une erreur est détectée lors du remplacement du rappel, la fonction est désenregistrée.

Si un rappel est enregistré dans la même fonction de rappel dans laquelle il a été précédemment désenregistré, il est traité comme une opération de remplacement ; les appels initiaux ou finaux ne sont pas appelés.

Vous pouvez utiliser MQOP\_REGISTER avec MQOP\_SUSPEND ou MQOP\_RESUME.

#### **MQOP\_DÉSENREGISTREMENT**

Arrête la consommation de messages pour le descripteur d'objet et supprime le descripteur de ceux qui sont éligibles pour un rappel.

Un rappel est automatiquement désenregistré si le descripteur associé est fermé.

Si MQOP\_REGISTER est appelé à partir d'un consommateur et qu'un appel d'arrêt est défini pour le rappel, il est appelé lors du retour du consommateur.

Si cette opération est émise sur un *Hobj* sans consommateur enregistré, l'appel est renvoyé avec MQRC\_CALLBACK\_NOT\_REGISTERED.

#### **MQOP\_SUSPENSION**

Interrompt la consommation de messages pour le descripteur d'objet.

Si cette opération est appliquée à un gestionnaire d'événements, le gestionnaire d'événements n'obtient pas d'événements lorsqu'il est suspendu et les événements manqués lorsqu'il est à l'état suspendu ne sont pas fournis à l'opération lors de sa reprise.

En cas d'interruption, la fonction de consommateur continue d'obtenir les rappels de type de contrôle.

#### **MQOP\_RESUME**

Reprenez la consommation de messages pour le descripteur d'objet.

Si cette opération est appliquée à un gestionnaire d'événements, le gestionnaire d'événements n'obtient pas d'événements lorsqu'il est suspendu et les événements manqués lorsqu'il est à l'état suspendu ne sont pas fournis à l'opération lors de sa reprise.

### CallbackDesc

Type: MQCBD-entrée

Il s'agit d'une structure qui identifie la fonction de rappel qui est enregistrée par l'application et les options utilisées lors de son enregistrement.

Pour plus de détails sur la structure, voir [MQCBD](#) .

Le descripteur de rappel est requis uniquement pour l'option MQOP\_REGISTER ; si le descripteur n'est pas requis, l'adresse de paramètre transmise peut être null.

## HOBJ

Type : MQHOBJ - entrée

Ce descripteur représente l'accès qui a été établi à l'objet à partir duquel un message doit être consommé. Il s'agit d'un descripteur qui a été renvoyé à partir d'un précédent appel MQOPEN ou MQSUB (dans le paramètre **Hobj** ).

*Hobj* n'est pas requis lors de la définition d'une routine de gestionnaire d'événements (MQCBT\_EVENT\_HANDLER) et doit être spécifié en tant que MQHO\_NONE.

Si *Hobj* a été renvoyé par un appel MQOPEN, la file d'attente doit avoir été ouverte avec une ou plusieurs des options suivantes:

- MQOO\_INPUT\_SHARED
- MQOO\_INPUT\_EXCLUSIVE
- MQOO\_INPUT\_AS\_Q\_DEF
- MQOO\_BROWSE

## MsgDesc

Type: MQMD-entrée

Cette structure décrit les attributs du message requis ainsi que ceux du message récupéré.

Le paramètre **MsgDesc** définit les attributs des messages requis par le consommateur et la version du MQMD à transmettre au consommateur de message.

*MsgId*, *CorrelId*, *GroupId*, *MsgSeqNumber* et *Offset* dans le MQMD sont utilisés pour la sélection des messages, en fonction des options spécifiées dans le paramètre **GetMsgOpts** .

*Encoding* et *CodedCharSetId* sont utilisés pour la conversion de message si vous spécifiez l'option MQGMO\_CONVERT.

Pour plus de détails, voir MQMD .

*MsgDesc* est utilisé pour MQOP\_REGISTER et si vous avez besoin de valeurs autres que celles par défaut pour les zones. *MsgDesc* n'est pas utilisé pour un gestionnaire d'événements.

Si le descripteur n'est pas requis, l'adresse de paramètre transmise peut être null.

Notez que si plusieurs consommateurs sont enregistrés dans la même file d'attente avec des sélecteurs qui se chevauchent, le consommateur choisi pour chaque message n'est pas défini.

## GetMsgOpts

Type: MQGMO-input

Le paramètre **GetMsgOpts** contrôle la façon dont le consommateur de message obtient les messages. Toutes les options de ce paramètre ont la signification décrite dans «Options MQGMO-Get-message», à la page 382, lorsqu'elles sont utilisées sur un appel MQGET, sauf:

### MQGMO\_SET\_SIGNAL

Cette option n'est pas autorisée.

### MQGMO\_BROWSE\_FIRST, MQGMO\_BROWSE\_NEXT, MQGMO\_MARK\_\*

L'ordre des messages distribués à un consommateur de navigation est dicté par les combinaisons de ces options. Les combinaisons significatives sont les suivantes:

#### MQGMO\_BROWSE\_FIRST

Le premier message de la file d'attente est distribué de manière répétée au consommateur. Cela est utile lorsque le consommateur consomme de façon destructive le message dans le rappel. Utilisez cette option avec précaution.

#### MQGMO\_BROWSE\_NEXT

Le destinataire reçoit chaque message de la file d'attente, de la position actuelle du curseur jusqu'à la fin de la file d'attente.

**MQGMO\_BROWSE\_FIRST + MQGMO\_BROWSE\_NEXT**

Le curseur est réinitialisé au début de la file d'attente. Le destinataire reçoit ensuite chaque message jusqu'à ce que le curseur atteigne la fin de la file d'attente.

**MQGMO\_BROWSE\_FIRST + MQGMO\_MARK\_\***

A partir du début de la file d'attente, le consommateur reçoit le premier message non marqué de la file d'attente, qui est ensuite marqué pour ce consommateur. Cette combinaison permet de s'assurer que le consommateur peut recevoir de nouveaux messages ajoutés derrière le point de curseur en cours.

**MQGMO\_BROWSE\_NEXT + MQGMO\_MARK\_\***

A partir de la position du curseur, le consommateur reçoit le prochain message non marqué dans la file d'attente, qui est ensuite marqué pour ce consommateur. Utilisez cette combinaison avec précaution car des messages peuvent être ajoutés à la file d'attente derrière la position actuelle du curseur.

**MQGMO\_BROWSE\_FIRST + MQGMO\_BROWSE\_NEXT + MQGMO\_MARK\_\***

Cette combinaison n'est pas autorisée. S'il est utilisé, l'appel renvoie MQRC\_OPTIONS\_ERROR.

**MQGMO\_NO\_WAIT, MQGMO\_WAIT et WaitInterval**

Ces options contrôlent la façon dont le consommateur est appelé.

**MQGMO\_NO\_WAIT**

Le consommateur n'est jamais appelé avec MQRC\_NO\_MSG\_AVAILABLE. Le consommateur n'est appelé que pour les messages et les événements.

**MQGMO\_WAIT avec la valeur zéro WaitInterval**

Le code MQRC\_NO\_MSG\_AVAILABLE est transmis au destinataire lorsqu'aucun message n'est disponible et que le destinataire a été démarré ou qu'il a reçu au moins un message depuis le dernier code anomalie "aucun message".

Cela empêche le consommateur de s'interroger dans une boucle occupée lorsqu'un intervalle d'attente de zéro est spécifié.

**MQGMO\_WAIT et un WaitInterval positif**

Le consommateur est appelé après l'intervalle d'attente spécifié avec le code anomalie MQRC\_NO\_MSG\_AVAILABLE. Cet appel est effectué indépendamment du fait que des messages aient été distribués ou non au consommateur. Cela permet à l'utilisateur d'effectuer un traitement de signal de présence ou de type de lot.

**MQGMO\_WAIT et WaitInterval de MQWI\_UNLIMITED**

Indique une attente infinie avant de renvoyer MQRC\_NO\_MSG\_AVAILABLE. Le consommateur n'est jamais appelé avec MQRC\_NO\_MSG\_AVAILABLE.

*GetMsgOpts* est utilisé uniquement pour MQOP\_REGISTER et si vous avez besoin de valeurs autres que la valeur par défaut pour les zones. *GetMsgOpts* n'est pas utilisé pour un gestionnaire d'événements.

Si les *GetMsgOpts* ne sont pas obligatoires, l'adresse de paramètre transmise peut être null. L'utilisation de ce paramètre revient à spécifier MQGMO\_DEFAULT avec MQGMO\_FAIL\_IF QUIESCING.

Si un descripteur de propriétés de message est fourni dans la structure MQGMO, une copie est fournie dans la structure MQGMO qui est transmise au rappel de consommateur. En cas de retour de l'appel MQCB, l'application peut supprimer le descripteur de propriétés de message.

**CompCode**

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

**MQCC\_OK**

Achèvement réussi.

**MQCC\_WARNING**

Avertissement (achèvement partiel).

**MQCC\_FAILED**

Echec de l'appel.

**Motif**

Type : MQLONG - sortie

Les codes anomalie de la liste suivante sont ceux que le gestionnaire de files d'attente peut renvoyer pour le paramètre **Reason** .

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') Adaptateur non disponible.

**MQRC\_ADAPTER\_CONV\_LOAD\_ERROR**

(2133, X'855') Impossible de charger les modules de service de conversion de données.

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

**MQRC\_API\_EXIT\_ERROR**

(2374, X'946') Echec de l'exit API.

**MQRC\_API\_EXIT\_LOAD\_ERROR**

(2183, X'887') Impossible de charger l'exit API.

**MQRC\_ASID\_MISMATCH**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

**MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

**MQRC\_CALLBACK\_LINK\_ERROR**

(2487, X'9B7') Zone de type de rappel incorrecte.

**MQRC\_CALLBACK\_NOT\_REGISTERED**

(2448, X' 990') Impossible de désenregistrer, de suspendre ou de reprendre car il n'y a pas de rappel enregistré.

**MQRC\_CALLBACK\_ROUTINE\_ERROR**

(2486, X'9B6') *CallbackFunction* ou *CallbackName* doit être spécifié, mais pas les deux.

**MQRC\_CALLBACK\_TYPE\_ERREUR**

(2483, X'9B3') Zone de type de rappel incorrecte.

**MQRC\_CBD\_OPTIONS\_ERREUR**

(2484, X'9B4') Zone d'options MQCBD incorrecte.

**MQRC\_CICS\_WAIT\_FAILED**

(2140, X'85C') Demande d'attente rejetée par CICS.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**MQRC\_CONNECTION\_NOT\_AUTHORIZED**

(2217, X'8A9') Non autorisé pour la connexion.

**MQRC\_CONNECTION QUIESCING**

(2202, X'89A') Connexion en cours de mise au repos.

**MQRC\_CONNECTION\_STOPPING**

(2203, X'89B') Connexion en cours d'arrêt.

**MQRC\_CORREL\_ID\_ERROR**

(2207, X'89F') Erreur d'identificateur de corrélation.



**MQRC\_DATA\_LENGTH\_ERROR**  
(2010, X'7DA') Paramètre de longueur des données non valide.

**MQRC\_ENVIRONMENT\_ERROR**  
(2012, X'7DC') Appel non valide dans l'environnement.

**MQRC\_FUNCTION\_NOT\_SUPPORTED**  
(2298, X'8FA') La fonction demandée n'est pas disponible dans l'environnement en cours.

**MQRC\_GET\_INHIBITED**  
(2016, X'7E0') Se bloque pour la file d'attente.

**MQRC\_GLOBAL\_UOW\_CONFLICT**  
(2351, X'92F') Conflit entre les unités d'oeuvre globales.

**MQRC\_GMO\_ERROR**  
(2186, X'88A') Structure des options de message Get non valide.

**MQRC\_HANDLE\_IN\_USE\_FOR\_UOW**  
(2353, X'931') Descripteur en cours d'utilisation pour l'unité d'oeuvre globale.

**MQRC\_HCONN\_ERROR**  
(2018, X'7E2') Descripteur de connexion non valide.

**MQRC\_HOBJ\_ERROR**  
(2019, X'7E3') Descripteur d'objet non valide.

**MQRC\_INCONSISTENT\_BROWSE**  
(2259, X'8D3') Spécification de navigation incohérente.

**MQRC\_INCONSISTENT\_UOW**  
(2245, X'8C5') Spécification d'unité d'oeuvre incohérente.

**MQRC\_INVALID\_MSG\_UNDER\_CURSOR**  
(2246, X'8C6') Message sous le curseur non valide pour la récupération.

**MQRC\_LOCAL\_UOW\_CONFLICT**  
(2352, X'930') Conflit entre l'unité d'oeuvre globale et l'unité d'oeuvre locale.

**MQRC\_MATCH\_OPTIONS\_ERROR**  
(2247, X'8C7') Options de correspondance non valides.

**MQRC\_MAX\_MSG\_LENGTH\_ERROR**  
(2485, X'9B4') Zone *MaxMsgLength* incorrecte.

**MQRC\_MD\_ERROR**  
(2026, X'7EA') Descripteur de message non valide.

**MQRC\_MODULE\_ENTRY\_NOT\_FOUND**  
(2497, X'9C1') Le point d'entrée de fonction spécifié est introuvable dans le module.

**MQRC\_MODULE\_INVALID**  
(2496, X'9C0') Le module a été trouvé, mais son type est incorrect ; il ne s'agit pas d'un module 32 bits, 64 bits ou d'une bibliothèque de liens dynamiques valide.

**MQRC\_MODULE\_NOT\_FOUND**  
(2495, X'9BF') Module introuvable dans le chemin de recherche ou non autorisé à être chargé.

**MQRC\_MSG\_SEQ\_NUMBER\_ERROR**  
(2250, X'8CA') Numéro de séquence de message non valide.

**MQRC\_MSG\_TOKEN\_ERROR**  
(2331, X'91B') Utilisation du jeton de message non valide.

**MQRC\_NO\_MSG\_AVAILABLE**  
(2033, X'7F1') Aucun message disponible.

**MQRC\_NO\_MSG\_UNDER\_CURSOR**  
(2034, X'7F2') Curseur d'exploration non positionné sur le message.

**MQRC\_NOT\_OPEN\_FOR\_BROWSE**  
(2036, X'7F4') File d'attente non ouverte pour exploration.

**MQRC\_NOT\_OPEN\_FOR\_INPUT**  
(2037, X'7F5') File d'attente non ouverte pour saisie.

**MQRC\_OBJECT\_CHANGED**  
(2041, X'7F9') Définition d'objet modifiée depuis son ouverture.

**MQRC\_OBJECT\_DAMAGED**  
(2101, X'835') Objet endommagé.

**ERREUR D'OPERATION\_MQRC**  
(2206, X'89E') Code d'opération incorrect sur l'appel API.

**MQRC\_OPTIONS\_ERROR**  
(2046, X'7FE') Options incorrectes ou incohérentes.

**MQRC\_PAGESET\_ERROR**  
(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.

**MQRC\_Q\_DELETED**  
(2052, X'804') File d'attente supprimée.

**MQRC\_Q\_INDEX\_TYPE\_ERROR**  
(2394, X'95A') Type d'index incorrect pour la file d'attente.

**MQRC\_Q\_MGR\_NAME\_ERROR**  
(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**  
(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

**MQRC\_Q\_MGR QUIESCING**  
(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.

**MQRC\_Q\_MGR\_STOPPING**  
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**MQRC\_RESOURCE\_PROBLEM**  
(2102, X'836') Ressources système disponibles insuffisantes.

**MQRC\_SIGNAL\_OUTSTANDING**  
(2069, X'815') Signal en attente pour ce descripteur.

**MQRC\_STORAGE\_NOT\_AVAILABLE**  
(2071, X'817') Mémoire disponible insuffisante.

**MQRC\_SUPPRESSED\_BY\_EXIT**  
(2109, X'83D') Appel supprimé par le programme d'exit.

**MQRC\_SYNCPOINT\_LIMIT\_REACHED**  
(2024, X'7E8') Aucun autre message pouvant être traité au sein de l'unité d'oeuvre en cours.

**MQRC\_SYNCPOINT\_NOT\_AVAILABLE**  
(2072, X'818') Prise en charge du point de synchronisation non disponible.

**MQRC\_UNEXPECTED\_ERROR**  
(2195, X'893') Une erreur inattendue s'est produite.

**MQRC\_UOW\_ENLISTMENT\_ERROR**  
(2354, X'932') Echec de l'inscription dans l'unité d'oeuvre globale.

**MQRC\_UOW\_MIX\_NOT\_SUPPORTED**  
(2355, X'933') Mélange d'appels d'unité d'oeuvre non pris en charge.

**MQRC\_UOW\_NOT\_AVAILABLE**  
(2255, X'8CF') Unité d'oeuvre non disponible pour le gestionnaire de files d'attente à utiliser.

**MQRC\_WAIT\_INTERVAL\_ERROR**  
(2090, X'82A') Intervalle d'attente non valide dans MQGMO.

**MQRC\_WRONG\_GMO\_VERSION**  
(2256, X'8D0') Version MQGMO fournie non valide.

**MQRC\_WRONG\_MD\_VERSION**  
(2257, X'8D1') Version MQMD fournie non valide.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Remarques sur l'utilisation

1. MQCB permet de définir l'action à appeler pour chaque message, correspondant aux critères spécifiés, disponible dans la file d'attente. Lorsque l'action est traitée, le message est supprimé de la file d'attente et transmis au consommateur de message défini, ou un jeton de message est fourni, qui est utilisé pour extraire le message.
2. MQCB peut être utilisé pour définir des routines de rappel avant de démarrer la consommation avec MQCTL ou à partir d'une routine de rappel.
3. Pour utiliser MQCB en dehors d'une routine de rappel, vous devez d'abord interrompre la consommation des messages à l'aide de MQCTL et reprendre la consommation par la suite.
4. MQCB n'est pas pris en charge dans l'adaptateur IMS .

## Séquence de rappel de consommateur de message

Vous pouvez configurer un destinataire pour qu'il appelle le rappel à des points clés au cours du cycle de vie du destinataire. Exemple :

- lorsque le consommateur est enregistré pour la première fois,
- lorsque la connexion est démarrée,
- lorsque la connexion est arrêtée et
- lorsque le consommateur est désenregistré, soit explicitement, soit implicitement par un MQCLOSE.

Verbe	Explication
MQCTL (DEMARRAGE)	Appel MQCTL à l'aide de l'opération MQOP_START
MQCTL (STOP)	Appel MQCTL à l'aide de l'opération MQOP_STOP
MQCTL (en attente)	Appel MQCTL à l'aide de l'opération MQOP_START_WAIT

Cela permet au consommateur de conserver l'état associé au consommateur. Lorsqu'un rappel est demandé par une application, les règles d'appel du consommateur sont les suivantes:

### S'INSCRIRE

Est toujours le premier type d'appel du rappel.

Est toujours appelé sur la même unité d'exécution que l'appel MQCB (REGISTER).

### DEBUT

Est toujours appelé de manière synchrone avec l'instruction MQCTL (START).

- Tous les rappels START sont terminés avant que l'instruction MQCTL (START) ne soit renvoyée.

Se trouve sur la même unité d'exécution que la distribution des messages si THREAD\_AFFINITY est demandé.

L'appel avec démarrage n'est pas garanti si, par exemple, un rappel précédent émet MQCTL (STOP) pendant MQCTL (START).

### ARRETER

Aucun autre message ou événement n'est distribué après cet appel tant que la connexion n'est pas redémarrée.

Une commande STOP est garantie si l'application a été précédemment appelée pour la commande START, un message ou un événement.

### ANNULER l'enregistrement

Est toujours le dernier type d'appel du rappel.

Assurez-vous que votre application effectue une initialisation et un nettoyage basés sur les unités d'exécution dans les rappels START et STOP. Vous pouvez effectuer une initialisation et un nettoyage non basés sur les unités d'exécution avec les rappels REGISTER et REGISISTER.

Ne faites aucune hypothèse sur la durée de vie et la disponibilité de l'unité d'exécution autre que celle qui est indiquée. Par exemple, ne vous fiez pas à une unité d'exécution qui reste active après le dernier appel à REGISTERISTER. De même, lorsque vous avez choisi de ne pas utiliser THREAD\_AFFINITY, ne supposez pas que l'unité d'exécution existe à chaque fois que la connexion est démarrée.

Si votre application a des exigences particulières en matière de caractéristiques d'unité d'exécution, elle peut toujours créer une unité d'exécution en conséquence, puis utiliser MQCTL (WAIT). Cela a pour effet de donner l'unité d'exécution à IBM MQ pour la distribution asynchrone des messages.

## Utilisation de la connexion de consommateur de message

Vous pouvez configurer un destinataire pour qu'il appelle le rappel à des points clés au cours du cycle de vie du destinataire. Exemple :

- lorsque le consommateur est enregistré pour la première fois,
- lorsque la connexion est démarrée,
- lorsque la connexion est arrêtée et
- lorsque le consommateur est désenregistré, soit explicitement, soit implicitement par un MQCLOSE.

Tableau 542. Définitions d'instruction MQCTL	
Verbe	Explication
MQCTL (DEMARRAGE)	Appel MQCTL à l'aide de l'opération MQOP_START
MQCTL (STOP)	Appel MQCTL à l'aide de l'opération MQOP_STOP
MQCTL (en attente)	Appel MQCTL à l'aide de l'opération MQOP_START_WAIT

Cela permet au consommateur de conserver l'état associé au consommateur. Lorsqu'un rappel est demandé par une application, les règles d'appel du consommateur sont les suivantes:

### S'INSCRIRE

Est toujours le premier type d'appel du rappel.

Est toujours appelé sur la même unité d'exécution que l'appel MQCB (REGISTER).

### DEBUT

Est toujours appelé de manière synchrone avec l'instruction MQCTL (START).

- Tous les rappels START sont terminés avant que l'instruction MQCTL (START) ne soit renvoyée.

Se trouve sur la même unité d'exécution que la distribution des messages si THREAD\_AFFINITY est demandé.

L'appel avec démarrage n'est pas garanti si, par exemple, un rappel précédent émet MQCTL (STOP) pendant MQCTL (START).

### ARRETER

Aucun autre message ou événement n'est distribué après cet appel tant que la connexion n'est pas redémarrée.

Une commande STOP est garantie si l'application a été précédemment appelée pour la commande START, un message ou un événement.

### ANNULER l'enregistrement

Est toujours le dernier type d'appel du rappel.

Assurez-vous que votre application effectue une initialisation et un nettoyage basés sur les unités d'exécution dans les rappels START et STOP. Vous pouvez effectuer une initialisation et un nettoyage non basés sur les unités d'exécution avec les rappels REGISTER et REGISISTER.

Ne faites aucune hypothèse sur la durée de vie et la disponibilité de l'unité d'exécution autre que celle qui est indiquée. Par exemple, ne vous fiez pas à une unité d'exécution qui reste active après le dernier appel à REGISTER. De même, lorsque vous avez choisi de ne pas utiliser THREAD\_AFFINITY, ne supposez pas que l'unité d'exécution existe à chaque fois que la connexion est démarrée.

Si votre application a des exigences particulières en matière de caractéristiques d'unité d'exécution, elle peut toujours créer une unité d'exécution en conséquence, puis utiliser MQCTL (WAIT). Cela a pour effet de donner l'unité d'exécution à IBM MQ pour la distribution asynchrone des messages.

## Appel C

```
MQCB (Hconn, Operation, CallbackDesc, Hobj, MsgDesc,
      GetMsgOpts, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;          /* Connection handle */
MQLONG   Operation;     /* Operation being processed */
MQCBD    CallbackDesc;  /* Callback descriptor */
MQHOBJ   HObj;          /* Object handle */
MQMD     MsgDesc        /* Message descriptor attributes */
MQGMO    GetMsgOpts     /* Message options */
MQLONG   CompCode;     /* Completion code */
MQLONG   Reason;       /* Reason code qualifying CompCode */
```

## Appel COBOL

```
CALL 'MQCB' USING HCONN, OPERATION, CBDESC, HOBJ, MSGDESC,
                GETMSGOPTS, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle
01 HCONN PIC S9(9) BINARY.
** Operation
01 OPERATION PIC S9(9) BINARY.
** Callback Descriptor
01 CBDESC.
   COPY CMQCBDV.
01 HOBJ PIC S9(9) BINARY.
** Message Descriptor
01 MSGDESC.
   COPY CMQMDV.
** Get Message Options
01 GETMSGOPTS.
   COPY CMQGMV.
** Completion code
01 COMPCODE PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON PIC S9(9) BINARY.
```

## Appel PL/I

```
call MQCB(Hconn, Operation, CallbackDesc, Hobj, MsgDesc, GetMsgOpts,
          CompCode, Reason)
```

Déclarez les paramètres comme suit :

```
dcl Hconn      fixed bin(31); /* Connection handle */
dcl Operation  fixed bin(31); /* Operation */
dcl CallbackDesc like MQCBD; /* Callback Descriptor */
dcl Hobj       fixed bin(31); /* Object Handle */
dcl MsgDesc    like MQMD; /* Message Descriptor */
```

```
dc1 GetMsgOpts    like MQGMO;      /* Get Message Options */
dc1 CompCode     fixed bin(31); /* Completion code */
dc1 Reason       fixed bin(31); /* Reason code qualifying CompCode */
```

## MQCB\_FUNCTION-Fonction de rappel

L'appel de fonction MQCB\_FUNCTION est la fonction de rappel pour la gestion des événements et la consommation des messages asynchrones.

La définition d'appel MQCB\_FUNCTION est fournie uniquement pour décrire les paramètres qui sont transmis à la fonction de rappel. Aucun point d'entrée appelé MQCB\_FUNCTION n'est fourni par le gestionnaire de files d'attente.

La spécification de la fonction réelle à appeler est une entrée de l'appel [MQCB](#) et est transmise via la structure [MQCBD](#).

### Syntaxe

MQCB\_FUNCTION (*Hconn*, *MsgDesc*, *GetMsgOpts*, *Buffer*, *Contexte*)

### Paramètres

#### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent. Sur les applications z/OS for CICS, l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour Hconn:

#### MQHC\_DEF\_CONN

Descripteur de connexion par défaut.

#### MsgDesc

Type: MQMD-entrée

Cette structure décrit les attributs du message extrait.

Pour plus d'informations, voir «MQMD-Descripteur de message», à la page 438.

La version de MQMD transmise est la même que celle transmise sur l'appel MQCB qui a défini la fonction de consommateur.

L'adresse du MQMD est transmise sous forme de caractères nuls si un MQGMO version 4 a été utilisé pour demander qu'un descripteur de message soit renvoyé à la place d'un MQMD.

Il s'agit d'une zone d'entrée de la fonction de consommateur de message ; elle n'est pas pertinente pour une fonction de gestionnaire d'événements.

#### GetMsgOpts

Type: MQGMO-input

Options utilisées pour contrôler les actions du consommateur de message. Ce paramètre contient également des informations supplémentaires sur le message renvoyé.

Pour plus d'informations, voir [MQGMO](#).

La version de MQGMO transmise est la version la plus récente prise en charge.

Il s'agit d'une zone d'entrée de la fonction de consommateur de message ; elle n'est pas pertinente pour une fonction de gestionnaire d'événements.

#### Mémoire tampon

Type: MQBYTEExBufferLength-input

Il s'agit de la zone contenant les données de message.

Si aucun message n'est disponible pour cet appel ou si le message ne contient aucune donnée de message, l'adresse du *Buffer* est transmise sous la forme de valeurs NULL.

Il s'agit d'une zone d'entrée de la fonction de consommateur de message ; elle n'est pas pertinente pour une fonction de gestionnaire d'événements.

### Contexte

Type: MQCBC-entrée / sortie

Cette structure fournit des informations de contexte aux fonctions de rappel. Pour plus d'informations, voir «MQCBC-Contexte de rappel», à la page 288.

### Remarques sur l'utilisation

1. Sachez que si vos routines de rappel utilisent des services qui peuvent retarder ou bloquer l'unité d'exécution, par exemple, MQGET avec attente, peut retarder la répartition d'autres rappels.
2. Une unité de travail distincte n'est pas automatiquement établie pour chaque appel d'une routine de rappel. Par conséquent, les routines peuvent émettre un appel de validation ou différer la validation jusqu'à ce qu'un lot logique de travail ait été traité. Lorsque le lot de travaux est validé, il valide les messages pour toutes les fonctions de rappel qui ont été appelées depuis le dernier point de synchronisation.
3. Les programmes appelés par CICS LINK ou CICS START extraient des paramètres à l'aide des services CICS via des objets nommés appelés conteneurs de canal. Les noms de conteneur sont identiques aux noms de paramètre. Pour plus d'informations, reportez-vous à votre documentation CICS.
4. Les routines de rappel peuvent émettre un appel MQDISC, mais pas pour leur propre connexion. Par exemple, si une routine de rappel a créé une connexion, elle peut également la déconnecter.
5. En règle générale, une routine de rappel ne doit pas être appelée à chaque fois à partir de la même unité d'exécution. Si nécessaire, utilisez MQCTLO\_THREAD\_AFFINITY lorsque la connexion est démarrée.
6. Lorsqu'une routine de rappel reçoit un code anomalie différent de zéro, elle doit effectuer l'action appropriée.
7. MQCB\_FUNCTION n'est pas pris en charge dans l'adaptateur IMS .

### MQCLOSE-Fermer l'objet

L'appel MQCLOSE abandonne l'accès à un objet et est l'inverse des appels MQOPEN et MQSUB.

### Syntaxe

MQCLOSE (*Hconn*, *Hobj*, *Options*, *CompCode*, *Raison*)

### Paramètres

#### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS , vous pouvez omettre l'appel MQCONN et spécifier la valeur suivante pour *Hconn*:

#### MQHC\_DEF\_HCONN

Descripteur de connexion par défaut.

#### HOBJ

Type: MQHOBJ-entrée / sortie

Ce descripteur représente l'objet en cours de fermeture. L'objet peut être de n'importe quel type. La valeur de *Hobj* a été renvoyée par un appel MQOPEN précédent.

Une fois l'appel terminé, le gestionnaire de files d'attente définit ce paramètre sur une valeur qui n'est pas un descripteur valide pour l'environnement. Cette valeur est:

### **MQHO\_UNUSABLE\_HOBJ**

Descripteur d'objet inutilisable.

Sous z/OS, *Hobj* est défini sur une valeur non définie.

## **Options**

Type : MQLONG - entrée

Ce paramètre contrôle la façon dont l'objet est fermé.

Seules les files d'attente dynamiques permanentes et les abonnements peuvent être fermés de plusieurs manières, car ils doivent être conservés ou supprimés ; il s'agit des files d'attente avec l'attribut **DefinitionType** dont la valeur est MQQDT\_PERMANENT\_DYNAMIC (voir l'attribut **DefinitionType** décrit dans «Attributs des files d'attente», à la page 875 ). Les options de fermeture sont résumées dans cette rubrique.

Les abonnements durables peuvent être conservés ou supprimés ; ils sont créés à l'aide de l'appel MQSUB avec l'option MQSO\_DURABLE.

Lors de la fermeture de l'indicateur sur une destination gérée (c'est-à-dire le paramètre **Hobj** renvoyé sur un appel MQSUB qui utilisait l'option MQSO\_MANAGED), le gestionnaire de files d'attente nettoie toutes les publications qui n'ont pas été extraites lorsque l'abonnement associé a également été supprimé. L'abonnement est supprimé à l'aide de l'option MQCO\_REMOVE\_SUB du paramètre **Hsub** renvoyé dans un appel MQSUB. Notez que MQCO\_REMOVE\_SUB est le comportement par défaut sur MQCLOSE pour un abonnement non durable.

Lors de la fermeture d'un descripteur vers une destination non gérée, vous êtes responsable du nettoyage de la file d'attente dans laquelle les publications sont envoyées. Fermez d'abord l'abonnement à l'aide de MQCO\_REMOVE\_SUB, puis traitez les messages hors de la file d'attente jusqu'à ce qu'il n'en reste plus.

Vous ne devez spécifier qu'une seule option parmi les suivantes:

**Options de file d'attente dynamique:** Ces options contrôlent la manière dont les files d'attente dynamiques permanentes sont fermées.

### **MQCO\_DELETE**

La file d'attente est supprimée si l'une des conditions suivantes est vérifiée:

- Il s'agit d'une file d'attente dynamique permanente, créée par un précédent appel MQOPEN, et il n'y a aucun message dans la file d'attente et aucune demande d'extraction ou d'insertion non validée en attente pour la file d'attente (pour la tâche en cours ou toute autre tâche).
- Il s'agit de la file d'attente dynamique temporaire créée par l'appel MQOPEN qui a renvoyé *Hobj*. Dans ce cas, tous les messages de la file d'attente sont purgés.

Dans tous les autres cas, y compris le cas où *Hobj* a été renvoyé sur un appel MQSUB, l'appel échoue avec le code anomalie MQRC\_OPTION\_NOT\_VALID\_FOR\_TYPE et l'objet n'est pas supprimé.

Sous z/OS, si la file d'attente est une file d'attente dynamique qui a été supprimée logiquement et qu'il s'agit de son dernier descripteur, la file d'attente est supprimée physiquement. Pour plus de détails, voir «Remarques sur l'utilisation», à la page 685.

### **MQCO\_DELETE\_PURGE**

La file d'attente est supprimée et tous les messages qu'elle contient sont purgés, si l'une des conditions suivantes est vérifiée:

- Il s'agit d'une file d'attente dynamique permanente, créée par un appel MQOPEN précédent, et aucune demande d'extraction ou d'insertion non validée n'est en attente pour la file d'attente (pour la tâche en cours ou toute autre tâche).
- Il s'agit de la file d'attente dynamique temporaire créée par l'appel MQOPEN qui a renvoyé *Hobj*.



Dans tous les autres cas, y compris le cas où *Hobj* a été renvoyé sur un appel MQSUB, l'appel échoue avec le code anomalie MQRC\_OPTION\_NOT\_VALID\_FOR\_TYPE et l'objet n'est pas supprimé.

*Tableau 543. Options de fermeture pour différents types d'objet*

Type d'objet ou de file d'attente	MQCO_AUCUN	MQCO_DELETE	MQCO_DELETE_PURGE
Objet autre qu'une file d'attente	Conservé	Non valide	Non valide
File d'attente prédéfinie	Conservé	Non valide	Non valide
file d'attente dynamique permanente	Conservé	Supprimé s'il est vide et qu'aucune mise à jour n'est en attente	Messages supprimés ; file d'attente supprimée si aucune mise à jour n'est en attente
File d'attente dynamique temporaire (appel émis par le créateur de la file d'attente)	Supprimé	Supprimé	Supprimé
File d'attente dynamique temporaire (appel non émis par le créateur de la file d'attente)	Conservé	Non valide	Non valide
Liste de distribution	Conservé	Non valide	Non valide
Destination d'abonnement géré	Conservé	Non valide	Non valide
Liste de distribution (l'abonnement a été supprimé)	Messages supprimés ; file d'attente supprimée	Non valide	Non valide

**Options de fermeture d'abonnement:** ces options contrôlent si les abonnements durables sont supprimés lorsque le descripteur est fermé et si les publications en attente de lecture par l'application sont nettoyées. Ces options ne peuvent être utilisées qu'avec un descripteur d'objet renvoyé dans le paramètre **Hsub** d'un appel MQSUB.

#### **MQCO\_KEEP\_SUB**

Le descripteur de l'abonnement est fermé, mais l'abonnement effectué est conservé. Les publications continuent d'être envoyées à la destination spécifiée dans l'abonnement. Cette option est valide uniquement si l'abonnement a été effectué avec l'option MQSO\_DURABLE.

MQCO\_KEEP\_SUB est la valeur par défaut si l'abonnement est durable

#### **SOUS-TITRE\_REMOVE\_MQQUE**

L'abonnement est supprimé et le descripteur de l'abonnement est fermé.

Le paramètre **Hobj** de l'appel MQSUB n'est pas invalidé par la fermeture du paramètre **Hsub** et peut continuer à être utilisé pour que MQGET ou MQCB reçoive les publications restantes. Lorsque le paramètre **Hobj** de l'appel MQSUB est également fermé, s'il s'agissait d'une destination gérée, les publications non extraites sont supprimées.

MQCO\_REMOVE\_SUB est la valeur par défaut si l'abonnement est non durable.

L'exécution réussie de MQCO\_REMOVE\_SUB ne signifie pas que l'action a abouti. Pour vérifier que cet appel est terminé, voir l'étape [DELETE SUB](#) dans [Vérification de la fin des commandes asynchrones pour les réseaux distribués](#).

Ces options de clôture d'abonnement sont récapitulées dans les tableaux suivants.

Tableau 544. Options permettant de fermer un descripteur d'abonnement durable mais de conserver l'abonnement

Tâche	Option de fermeture d'abonnement
Conserver les publications sur un descripteur MQOPENed	MQCO_KEEP_SUB
Suppression de publications sur un descripteur MQOPENed	Action non autorisée
Conserver les publications sur un descripteur MQSO_MANAGED	MQCO_KEEP_SUB
Suppression de publications sur un descripteur MQSO_MANAGED	Action non autorisée

Pour vous désabonner, en fermant un descripteur d'abonnement durable et en le désabonnant ou en fermant un descripteur d'abonnement non durable, utilisez les options de fermeture d'abonnement suivantes:

Tableau 545. Options de désabonnement

Tâche	Option de fermeture d'abonnement
Conserver les publications sur un descripteur MQOPENed	SOUS-TITRE_REMOVE_MQQUE
Suppression de publications sur un descripteur MQOPENed	Action non autorisée
Conserver les publications sur un descripteur MQSO_MANAGED	SOUS-TITRE_REMOVE_MQQUE

**Options de lecture anticipée:** Les options suivantes contrôlent ce qui arrive aux messages non persistants qui ont été envoyés au client avant qu'une application ne les demande et qui n'ont pas encore été consommés par l'application. Ces messages sont stockés dans la mémoire tampon de lecture anticipée du client en attente d'être demandés par l'application et peuvent être supprimés ou consommés dans la file d'attente avant la fin de l'opération MQCLOSE.

#### **MQCO\_IMMEDIATE**

L'objet est fermé immédiatement et tous les messages qui ont été envoyés au client avant qu'une application ne les demande sont supprimés et ne peuvent être consommés par aucune application. Il s'agit de la valeur par défaut.

#### **MQCO\_QUIESCE**

Une demande de fermeture de l'objet est effectuée, mais si des messages qui ont été envoyés au client avant qu'une application ne les demande, résident toujours dans la mémoire tampon de lecture anticipée du client, l'appel MQCLOSE est renvoyé avec un avertissement MQRC\_READ\_AHEAD\_MSGS et le descripteur d'objet reste valide.

L'application peut ensuite continuer à utiliser le descripteur d'objet pour extraire des messages jusqu'à ce qu'il ne soit plus disponible, puis fermer à nouveau l'objet. Aucun autre message n'est envoyé au client avant qu'une application ne le demande. La lecture anticipée est désormais désactivée.

Il est conseillé aux applications d'utiliser MQCO\_QUIESCE plutôt que d'essayer d'atteindre un point où il n'y a plus de messages dans la mémoire tampon de lecture anticipée du client, car un message pourrait arriver entre le dernier appel MQGET et le MQCLOSE suivant qui serait supprimé si MQCO\_IMMEDIATE était utilisé.

Si un MQCLOSE avec MQCO\_QUIESCE est émis à partir d'une fonction de rappel asynchrone, le même comportement de lecture des messages à l'avance s'applique. Si l'avertissement MQRC\_READ\_AHEAD\_MSGS est renvoyé, la fonction de rappel est appelée au moins une

fois de plus. Lorsque le dernier message restant ayant fait l'objet d'une lecture anticipée a été transmis à la fonction de rappel, la zone ConsumerFlags de MQCBC est définie sur MQCBCF\_READA\_BUFFER\_EMPTY.

**Option par défaut:** Si vous n'avez besoin d'aucune des options décrites précédemment, vous pouvez utiliser l'option suivante:

### **MQCO\_AUCUN**

Aucun traitement de fermeture facultatif n'est requis.

Cette valeur doit être spécifiée pour:

- Objets autres que des files d'attente
- Files d'attente prédéfinies
- Files d'attente dynamiques temporaires (mais uniquement dans les cas où *Hobj* n'est pas le descripteur renvoyé par l'appel MQOPEN qui a créé la file d'attente).
- Listes de diffusion

Dans tous les cas ci-dessus, l'objet est conservé et non supprimé.

Si cette option est spécifiée pour une file d'attente dynamique temporaire:

- La file d'attente est supprimée si elle a été créée par l'appel MQOPEN qui a renvoyé *Hobj* ; tous les messages qui se trouvent dans la file d'attente sont purgés.
- Dans tous les autres cas, la file d'attente (et tous les messages qu'elle contient) est conservée.

Si cette option est spécifiée pour une file d'attente dynamique permanente, la file d'attente est conservée et n'est pas supprimée.

Sous z/OS, si la file d'attente est une file d'attente dynamique qui a été supprimée logiquement et qu'il s'agit de son dernier descripteur, la file d'attente est supprimée physiquement. Pour plus de détails, voir «Remarques sur l'utilisation», à la page 685.

### **CompCode**

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_WARNING**

Avertissement (achèvement partiel).

#### **MQCC\_FAILED**

Echec de l'appel.

### **Motif**

Type : MQLONG - sortie

Les codes raison répertoriés sont ceux que le gestionnaire de files d'attente peut renvoyer pour le paramètre **Reason** .

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_WARNING:

#### **MQRC\_INCOMPLETE\_GROUP**

(2241, X'8C1') Groupe de messages non complet.

#### **MQRC\_INCOMPLETE\_MSG**

(2242, X'8C2') Message logique non complet.

**MQRC\_READ\_AHEAD\_MSGS**

(nnnn, X'xxx') Le client a lu des messages d'avance qui n'ont pas encore été consommés par l'application.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') Adaptateur non disponible.

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

**MQRC\_API\_EXIT\_ERROR**

(2374, X'946') Echec de l'exit API.

**MQRC\_API\_EXIT\_LOAD\_ERROR**

(2183, X'887') Impossible de charger l'exit API.

**MQRC\_ASID\_MISMATCH**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

**MQRC\_CF\_NOT\_XX\_ENCODE\_CASE\_ONE disponible**

(2345, X' 929') Unité de couplage non disponible.

**MQRC\_CF\_STRUC\_FAILED**

(2373, X'945') Echec de la structure de l'unité de couplage.

**MQRC\_CF\_STRUC\_IN\_USE**

(2346, X'92A') Structure de l'unité de couplage en cours d'utilisation.

**MQRC\_CICS\_WAIT\_FAILED**

(2140, X'85C') Demande d'attente rejetée par CICS.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**MQRC\_CONNECTION\_NOT\_AUTHORIZED**

(2217, X'8A9') Non autorisé pour la connexion.

**MQRC\_CONNECTION\_STOPPING**

(2203, X'89B') Connexion en cours d'arrêt.

**MQRC\_DB2\_NOT\_AVAILABLE**

(2342, X' 926') Sous-système Db2 non disponible.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') Descripteur de connexion non valide.

**MQRC\_HOBJ\_ERROR**

(2019, X'7E3') Descripteur d'objet non valide.

**MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') Non autorisé pour l'accès.

**MQRC\_OBJECT\_DAMAGED**

(2101, X'835') Objet endommagé.

**MQRC\_OPTION\_NOT\_VALID\_FOR\_TYPE**

(2045, X'7FD') Dans un appel MQOPEN ou MQCLOSE: option non valide pour le type d'objet.

**MQRC\_OPTIONS\_ERROR**

(2046, X'7FE') Options incorrectes ou incohérentes.

**MQRC\_PAGESET\_ERROR**

(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.

**MQRC\_Q\_MGR\_NAME\_ERROR**

(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**

(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**MQRC\_Q\_NOT\_EMPTY**

(2055, X'807') La file d'attente contient un ou plusieurs messages ou des demandes d'insertion ou d'extraction non validées.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') Ressources système disponibles insuffisantes.

**MQRC\_SECURITY\_ERROR (ERREUR DE SECURITE MQ)**

(2063, X'80F') Une erreur de sécurité s'est produite.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') Mémoire disponible insuffisante.

**MQRC\_SUPPRESSED\_BY\_EXIT**

(2109, X'83D') Appel supprimé par le programme d'exit.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Remarques sur l'utilisation

1. Lorsqu'une application émet l'appel MQDISC ou se termine normalement ou anormalement, tous les objets qui ont été ouverts par l'application et qui sont toujours ouverts sont fermés automatiquement avec l'option MQCO\_NONE.
2. Les points suivants s'appliquent si l'objet en cours de fermeture est une *file d'attente*:
  - Si les opérations sur la file d'attente sont effectuées dans le cadre d'une unité de travail, la file d'attente peut être fermée avant ou après l'exécution du point de synchronisation sans affecter le résultat du point de synchronisation. Si la file d'attente est déclenchée, l'exécution d'une annulation avant la fermeture de la file d'attente peut entraîner l'émission d'un message de déclenchement. Pour plus d'informations sur les messages de déclenchement, voir [Propriétés des messages de déclenchement](#).
  - Si la file d'attente a été ouverte avec l'option MQOO\_BROWSE, le curseur de navigation est détruit. Si la file d'attente est rouverte avec l'option MQOO\_BROWSE, un nouveau curseur de navigation est créé (voir [MQOO\\_BROWSE](#)).
  - Si un message est actuellement verrouillé pour ce descripteur au moment de l'appel MQCLOSE, le verrou est libéré (voir [MQGMO\\_LOCK](#)).
  - Sous z/OS, s'il existe une demande MQGET avec l'option MQGMO\_SET\_SIGNAL en attente sur le descripteur de file d'attente en cours de fermeture, la demande est annulée (voir [MQGMO\\_SET\\_SIGNAL](#)). Les demandes de signal pour la même file d'attente mais déposées sur des descripteurs différents (*Hobj*) ne sont pas affectées (sauf si une file d'attente dynamique est supprimée, auquel cas elles sont également annulées).
3. Les points suivants s'appliquent si l'objet en cours de fermeture est une *file d'attente dynamique* (permanente ou temporaire):
  - Pour une file d'attente dynamique, vous pouvez spécifier les options MQCO\_DELETE et MQCO\_DELETE\_PURGE, quelles que soient les options spécifiées dans l'appel MQOPEN correspondant.
  - Lorsqu'une file d'attente dynamique est supprimée, tous les appels MQGET avec l'option MQGMO\_WAIT qui sont en attente dans la file d'attente sont annulés et le code anomalie MQRC\_Q\_DELETED est renvoyé. Voir [MQGMO\\_WAIT](#).

Bien que les applications ne puissent pas accéder à une file d'attente supprimée, la file d'attente n'est pas supprimée du système et les ressources associées ne sont pas libérées tant que tous les

descripteurs qui font référence à la file d'attente n'ont pas été fermés et que toutes les unités de travail qui affectent la file d'attente n'ont pas été validées ou annulées.

Sous z/OS, une file d'attente qui a été logiquement supprimée mais qui n'a pas encore été supprimée du système empêche la création d'une nouvelle file d'attente portant le même nom que la file d'attente supprimée ; l'appel MQOPEN échoue avec le code anomalie MQRC\_NAME\_IN\_USE dans ce cas. De plus, une telle file d'attente peut toujours être affichée à l'aide des commandes MQSC, même si les applications ne peuvent pas y accéder.

- Lorsqu'une file d'attente dynamique permanente est supprimée, si le descripteur *Hobj* indiqué dans l'appel MQCLOSE n'est pas celui qui a été renvoyé par l'appel MQOPEN qui a créé la file d'attente, il est vérifié que l'ID utilisateur utilisé pour valider l'appel MQOPEN est autorisé à supprimer la file d'attente. Si l'option MQOO\_ALTERNATE\_USER\_AUTHORITY a été spécifiée sur l'appel MQOPEN, l'ID utilisateur vérifié est *AlternateUserId*.

Cette vérification n'est pas effectuée si:

- Le descripteur indiqué est celui renvoyé par l'appel MQOPEN qui a créé la file d'attente.
- La file d'attente en cours de suppression est une file d'attente dynamique temporaire.
- Lorsqu'une file d'attente dynamique temporaire est fermée, si le descripteur *Hobj* indiqué dans l'appel MQCLOSE est celui qui a été renvoyé par l'appel MQOPEN qui a créé la file d'attente, la file d'attente est supprimée. Cela se produit quelles que soient les options de fermeture spécifiées dans l'appel MQCLOSE. S'il y a des messages dans la file d'attente, ils sont supprimés ; aucun message de rapport n'est généré.

Si des unités de travail non validées affectent la file d'attente, la file d'attente et ses messages sont toujours supprimés, mais les unités de travail n'échouent pas. Toutefois, comme décrit précédemment, les ressources associées aux unités de travail ne sont pas libérées tant que chacune des unités de travail n'a pas été validée ou annulée.

4. Les points suivants s'appliquent si l'objet en cours de fermeture est une *liste de distribution*:

- La seule option de fermeture valide pour une liste de distribution est MQCO\_NONE ; l'appel échoue avec le code anomalie MQRC\_OPTIONS\_ERROR ou MQRC\_OPTION\_NOT\_VALID\_FOR\_TYPE si d'autres options sont spécifiées.
- Lorsqu'une liste de distribution est fermée, les codes achèvement et les codes raison individuels ne sont pas renvoyés pour les files d'attente de la liste ; seuls les paramètres **CompCode** et **Reason** de l'appel sont disponibles à des fins de diagnostic.

Si un incident se produit lors de la fermeture de l'une des files d'attente, le gestionnaire de files d'attente poursuit le traitement et tente de fermer les files d'attente restantes de la liste de distribution. Les paramètres **CompCode** et **Reason** de l'appel sont définis pour renvoyer des informations décrivant l'échec. Il est possible que le code achèvement soit MQCC\_FAILED, même si la plupart des files d'attente ont été fermées avec succès. La file d'attente qui a rencontré l'erreur n'est pas identifiée.

S'il y a un échec dans plusieurs files d'attente, il n'est pas défini quel échec est signalé dans les paramètres **CompCode** et **Reason** .

## Appel C

```
MQCLOSE (Hconn, &Hobj, Options, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;      /* Connection handle */
MQHOBJ   Hobj;      /* Object handle */
MQLONG   Options;   /* Options that control the action of MQCLOSE */
MQLONG   CompCode;  /* Completion code */
MQLONG   Reason;    /* Reason code qualifying CompCode */
```

## Appel COBOL

```
CALL 'MQCLOSE' USING HCONN, HOBJ, OPTIONS, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle
01 HCONN      PIC S9(9) BINARY.
** Object handle
01 HOBJ       PIC S9(9) BINARY.
** Options that control the action of MQCLOSE
01 OPTIONS    PIC S9(9) BINARY.
** Completion code
01 COMPCODE   PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON     PIC S9(9) BINARY.
```

## Appel PL/I

```
call MQCLOSE (Hconn, Hobj, Options, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
dcl Hconn      fixed bin(31); /* Connection handle */
dcl Hobj       fixed bin(31); /* Object handle */
dcl Options    fixed bin(31); /* Options that control the action of
                               MQCLOSE */
dcl CompCode   fixed bin(31); /* Completion code */
dcl Reason     fixed bin(31); /* Reason code qualifying CompCode */
```

## Appel d'assembleur de haut niveau

```
CALL MQCLOSE,(HCONN,HOBJ,OPTIONS,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
HCONN      DS F Connection handle
HOBJ       DS F Object handle
OPTIONS    DS F Options that control the action of MQCLOSE
COMPCODE   DS F Completion code
REASON     DS F Reason code qualifying COMPCODE
```

## Appel Visual Basic

```
MQCLOSE Hconn, Hobj, Options, CompCode, Reason
```

Déclarez les paramètres comme suit :

```
Dim Hconn      As Long 'Connection handle'
Dim Hobj       As Long 'Object handle'
Dim Options    As Long 'Options that control the action of MQCLOSE'
Dim CompCode   As Long 'Completion code'
Dim Reason     As Long 'Reason code qualifying CompCode'
```

## MQCMIT-Validation des modifications

L'appel MQCMIT indique au gestionnaire de files d'attente que l'application a atteint un point de synchronisation et que tous les messages d'obtention et d'insertion qui se sont produits depuis le dernier point de synchronisation doivent être rendus permanents.

Les messages insérés dans une unité de travail sont mis à la disposition d'autres applications ; les messages extraits dans le cadre d'une unité de travail sont supprimés.

- **z/OS** Sous z/OS, l'appel est utilisé uniquement par les programmes de traitement par lots (y compris les programmes DL/I de traitement par lots IMS).

### Syntaxe

MQCMIT (*Hconn*, *CompCode*, *Raison*)

### Paramètres

#### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

#### CompCode

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

##### **MQCC\_OK**

Achèvement réussi.

##### **MQCC\_WARNING**

Avertissement (achèvement partiel).

##### **MQCC\_FAILED**

Echec de l'appel.

#### Motif

Type : MQLONG - sortie

Les codes raison répertoriés sont ceux que le gestionnaire de files d'attente peut renvoyer pour le paramètre **Reason**.

Si *CompCode* est MQCC\_OK:

##### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_WARNING:

##### **MQRC\_BACKED\_OUT**

(2003, X'7D3') Unité d'oeuvre annulée.

##### **MQRC\_OUTCOME\_EN attente**

(2124, X'84C') Le résultat de l'opération de validation est en attente.

Si *CompCode* est MQCC\_FAILED:

##### **MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

##### **MQRC\_API\_EXIT\_ERROR**

(2374, X'946') Echec de l'exit API.

##### **MQRC\_ASID\_MISMATCH**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.



**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

**MQRC\_CALL\_INTERROMPUE**

(2549, X'9F5') MQPUT ou MQCMIT a été interrompu et le traitement de la reconnexion ne peut pas rétablir un résultat défini.

**MQRC\_CF\_STRUC\_IN\_USE**

(2346, X'92A') Structure de l'unité de couplage en cours d'utilisation.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**MQRC\_ENVIRONMENT\_ERROR**

(2012, X'7DC') Appel non valide dans l'environnement.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') Descripteur de connexion non valide.

**MQRC\_OBJECT\_DAMAGED**

(2101, X'835') Objet endommagé.

**MQRC\_OUTCOME\_MIXTE**

(2123, X'84B') Le résultat de l'opération de validation ou d'exclusion est mélangé.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**Echec de MQRC\_RECONNECT\_FAILED**

(2548, X'9F4') Après la reconnexion, une erreur s'est produite lors de la réinstallation des descripteurs d'une connexion reconnectable.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') Ressources système disponibles insuffisantes.

**MQRC\_STORAGE\_MEDIUM\_FULL**

(2192, X'890') Support de stockage externe saturé.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') Mémoire disponible insuffisante.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Remarques sur l'utilisation

1. Utilisez cet appel uniquement lorsque le gestionnaire de files d'attente lui-même coordonne l'unité d'oeuvre. Ce nom peut être :
  - Unité de travail locale, dans laquelle les modifications affectent uniquement les ressources IBM MQ .
  - Unité de travail globale dans laquelle les modifications peuvent affecter les ressources appartenant à d'autres gestionnaires de ressources, ainsi que les ressources IBM MQ .Pour plus de détails sur les unités d'oeuvre locales et globales, voir [«MQBEGIN-Début de l'unité de travail»](#), à la page 661.
2. Dans les environnements où le gestionnaire de files d'attente ne coordonne pas l'unité d'oeuvre, l'appel de validation approprié doit être utilisé à la place de MQCMIT. L'environnement peut également prendre en charge une validation implicite provoquée par l'arrêt normal de l'application.
  - Sous z/OS, utilisez les appels suivants:
    - Les programmes batch (y compris les programmes IMS batch DL/I) peuvent utiliser l'appel MQCMIT si l'unité de travail n'affecte que les ressources IBM MQ . Toutefois, si l'unité de travail affecte à la fois les ressources IBM MQ et les ressources appartenant à d'autres gestionnaires de ressources (par exemple, Db2 ), utilisez l'appel SRRCMIT fourni par le service RRS ( z/OS Recoverable Resource Service). L'appel SRRCMIT valide les modifications apportées

aux ressources appartenant aux gestionnaires de ressources qui ont été activés pour la coordination RRS.

- Les applications CICS doivent utiliser la commande EXEC CICS SYNCPOINT pour valider explicitement l'unité de travail. Sinon, l'arrêt de la transaction entraîne une validation implicite de l'unité d'oeuvre. L'appel MQCMIT ne peut pas être utilisé pour les applications CICS .
  - Les applications IMS (autres que les programmes batch DL/I) doivent utiliser des appels IMS tels que GU et CHKP pour valider l'unité de travail. L'appel MQCMIT ne peut pas être utilisé pour les applications IMS (autres que les programmes DL/I de traitement par lots).
  - Sous IBM i, utilisez cet appel pour les unités de travail locales coordonnées par le gestionnaire de files d'attente. Cela signifie qu'une définition de validation ne doit pas exister au niveau du travail, c'est-à-dire que la commande STRCMTCTL avec le paramètre **CMTSCOPE (\*JOB)** ne doit pas avoir été émise pour le travail.
3. Si une application se termine avec des modifications non validées dans une unité d'oeuvre, la disposition de ces modifications varie selon que l'application se termine normalement ou anormalement. Pour plus d'informations, voir [MQDISC usage notes](#) .
  4. Lorsqu'une application insère ou extrait des messages dans des groupes ou des segments de messages logiques, le gestionnaire de files d'attente conserve les informations relatives au groupe de messages et au message logique pour les derniers appels MQPUT et MQGET ayant abouti. Ces informations sont associées à l'identificateur de file d'attente et incluent les éléments suivants:
    - Valeurs des zones *GroupId*, *MsgSeqNumber*, *Offset* et *MsgFlags* dans MQMD.
    - Indique si le message fait partie d'une unité de travail.
    - Pour l'appel MQPUT: indique si le message est persistant ou non persistant.

Lorsqu'une unité d'oeuvre est validée, le gestionnaire de files d'attente conserve les informations de groupe et de segment et l'application peut continuer à insérer ou à extraire des messages dans le groupe de messages ou le message logique en cours.

La conservation des informations de groupe et de segment lors de la validation d'une unité de travail permet à l'application de répartir un grand groupe de messages ou un grand message logique composé de plusieurs segments sur plusieurs unités de travail. L'utilisation de plusieurs unités d'oeuvre est avantageuse si le gestionnaire de files d'attente local ne dispose que d'une mémoire de file d'attente limitée. Toutefois, l'application doit conserver suffisamment d'informations pour redémarrer l'insertion ou l'obtention de messages au point approprié en cas de défaillance du système. Pour plus d'informations sur le redémarrage au point approprié après une défaillance du système, voir [MQPMO\\_LOGICAL\\_ORDER](#) et [MQGMO\\_LOGICAL\\_ORDER](#).

Les autres remarques d'utilisation s'appliquent uniquement lorsque le gestionnaire de files d'attente coordonne les unités de travail:

5. Une unité de travail a la même portée qu'un descripteur de connexion ; tous les appels IBM MQ qui affectent une unité de travail particulière doivent être exécutés à l'aide du même descripteur de connexion. Les appels émis à l'aide d'un descripteur de connexion différent (par exemple, les appels émis par une autre application) affectent une unité d'oeuvre différente. Pour plus d'informations sur la portée des descripteurs de connexion, voir le paramètre **Hconn** décrit dans MQCONN.
6. Seuls les messages insérés ou extraits dans le cadre de l'unité d'oeuvre en cours sont affectés par cet appel.
7. Une application à exécution longue qui émet des appels MQGET, MQPUT ou MQPUT1 dans une unité de travail, mais qui n'émet jamais d'appel de validation ou d'exclusion, peut remplir les files d'attente avec des messages qui ne sont pas disponibles pour d'autres applications. Pour éviter cela, l'administrateur doit définir l'attribut de gestionnaire de files d'attente **MaxUncommittedMsgs** sur une valeur suffisamment faible pour empêcher les applications en boucle indéfinie de remplir les files d'attente, mais suffisamment élevée pour permettre aux applications de messagerie attendues de fonctionner correctement.

8. **ALW** Sur les systèmes AIX, Linux, and Windows , si le paramètre **Reason** est MQRC\_CONNECTION\_BROKEN (avec un *CompCode* de MQCC\_FAILED) ou MQRC\_UNEXPECTED\_ERROR, il est possible que l'unité de travail ait été validée avec succès.

## Appel C

```
MQCMIT (Hconn, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;      /* Connection handle */
MQLONG   CompCode;  /* Completion code */
MQLONG   Reason;    /* Reason code qualifying CompCode */
```

## Appel COBOL

```
CALL 'MQCMIT' USING HCONN, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle
01 HCONN    PIC S9(9) BINARY.
** Completion code
01 COMPCODE PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON   PIC S9(9) BINARY.
```

## Appel PL/I

```
call MQCMIT (Hconn, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
dcl Hconn      fixed bin(31); /* Connection handle */
dcl CompCode   fixed bin(31); /* Completion code */
dcl Reason     fixed bin(31); /* Reason code qualifying CompCode */
```

## Appel d'assembleur de haut niveau

```
CALL MQCMIT,(HCONN,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
HCONN      DS F Connection handle
COMPCODE   DS F Completion code
REASON     DS F Reason code qualifying COMPCODE
```

## Appel Visual Basic

```
MQCMIT Hconn, CompCode, Reason
```

Déclarez les paramètres comme suit :

```
Dim Hconn      As Long 'Connection handle'  
Dim CompCode  As Long 'Completion code'  
Dim Reason    As Long 'Reason code qualifying CompCode'
```

## MQCONN-Connexion du gestionnaire de files d'attente

L'appel MQCONN connecte un programme d'application à un gestionnaire de files d'attente.

Il fournit un descripteur de connexion de gestionnaire de files d'attente, que l'application utilise lors des appels de mise en file d'attente de messages ultérieurs.

- Sous z/OS, les applications CICS n'ont pas besoin d'émettre cet appel. Ces applications sont connectées automatiquement au gestionnaire de files d'attente auquel le système CICS est connecté. Toutefois, les appels MQCONN et MQDISC sont toujours acceptés à partir des applications CICS .
- Sous IBM i, les applications doivent utiliser l'appel MQCONN ou MQCONNX pour se connecter au gestionnaire de files d'attente et l'appel MQDISC pour se déconnecter du gestionnaire de files d'attente.

Une connexion client ne peut pas être établie sur une installation serveur uniquement et une connexion locale ne peut pas être établie sur une installation client uniquement.

### Syntaxe

MQCONN (*QMgrName*, *Hconn*, *CompCode*, *Motif*)

### Paramètres

#### QMgrName

Type: MQCHAR48 -entrée

Il s'agit du nom du gestionnaire de files d'attente auquel l'application souhaite se connecter. Le nom peut contenir les caractères suivants:

- Caractères alphabétiques en majuscules (A à Z)
- Caractères alphabétiques minuscules (a à z)
- Chiffres (0 à 9)
- Point (.), barre oblique (/), trait de soulignement (\_), pourcentage (%)

Le nom ne doit pas contenir d'espaces de début ou d'espaces imbriqués, mais peut contenir des espaces de fin. Un caractère NULL peut être utilisé pour indiquer la fin des données significatives dans le nom ; la valeur NULL et les caractères qui la suivent sont traités comme des blancs. Les restrictions suivantes s'appliquent dans les environnements indiqués:

- Sur les systèmes qui utilisent EBCDIC Katakana, les caractères minuscules ne peuvent pas être utilisés.
- Sous z/OS, les noms commençant ou se terminant par un trait de soulignement ne peuvent pas être traités par les panneaux d'opérations et de contrôle. Pour cette raison, évitez ces noms.
- Sous IBM i, les noms contenant des caractères minuscules, des barres obliques ou des pourcentages doivent être placés entre guillemets lorsqu'ils sont spécifiés dans les commandes. N'indiquez pas ces guillemets dans le paramètre **QMgrName** .

Si le nom est entièrement vide, le nom du gestionnaire de files d'attente *par défaut* est utilisé. Toutefois, notez l'utilisation de noms de gestionnaire de files d'attente vides décrits dans la section relative aux applications IBM MQ MQI client .

Le nom spécifié pour *QMgrName* doit être le nom d'un gestionnaire de files d'attente *connectable* ou, si des groupes de gestionnaires de files d'attente sont utilisés, le nom du groupe de gestionnaires de files d'attente.

Sous z/OS, les gestionnaires de files d'attente auxquels il est possible de se connecter sont déterminés par l'environnement:

- Pour CICS, vous pouvez utiliser uniquement le gestionnaire de files d'attente auquel le système CICS est connecté. Le paramètre **QMgrName** doit toujours être spécifié, mais sa valeur est ignorée ; les caractères blancs sont une option appropriée.
- Pour IMS, seuls les gestionnaires de files d'attente répertoriés dans la table de définition de sous-système (CSQQDEFV), et répertoriés dans la table SSM dans IMS, sont connectables (voir la note d'utilisation [6](#)).
- Pour z/OS batch et TSO, seuls les gestionnaires de files d'attente qui résident sur le même système que l'application sont connectables (voir la remarque d'utilisation [6](#)).

**Groupes de partage de files d'attente:** Sur les systèmes où plusieurs gestionnaires de files d'attente existent et sont configurés pour former un groupe de partage de files d'attente, le nom du groupe de partage de files d'attente peut être spécifié pour *QMgrName* à la place du nom d'un gestionnaire de files d'attente. Cela permet à l'application de se connecter à *tout gestionnaire de files d'attente* disponible dans le groupe de partage de files d'attente et qui se trouve sur la même image z/OS que l'application. Le système peut également être configuré de sorte que l'utilisation d'un *QMgrName* vide se connecte au groupe de partage de files d'attente au lieu du gestionnaire de files d'attente par défaut.

Si *QMgrName* spécifie le nom du groupe de partage de files d'attente, mais qu'il existe également un gestionnaire de files d'attente portant ce nom sur le système, la connexion est établie à ce dernier de préférence au premier. La connexion à l'un des gestionnaires de files d'attente du groupe de partage de files d'attente n'est tentée que si cette connexion échoue.

Si la connexion aboutit, vous pouvez utiliser le descripteur renvoyé par l'appel MQCONN ou MQCONNX pour accéder à *toutes* les ressources (partagées et non partagées) appartenant au gestionnaire de files d'attente auquel la connexion a été établie. L'accès à ces ressources est soumis aux contrôles d'autorisation standard.

Si l'application émet deux appels MQCONN ou MQCONNX pour établir des connexions simultanées et qu'un ou les deux appels spécifient le nom du groupe de partage de files d'attente, le deuxième appel renvoie le code achèvement MQCC\_WARNING et le code anomalie MQRC\_ALREADY\_CONNECTED lorsqu'il se connecte au même gestionnaire de files d'attente que le premier appel.

Les groupes de partage de files d'attente sont pris en charge uniquement sous z/OS. La connexion à un groupe de partage de files d'attente n'est prise en charge que dans les environnements par lots, RRS par lots, CICS et TSO. Pour CICS, vous pouvez utiliser uniquement le groupe de partage de files d'attente auquel le système CICS est connecté. Vous devez toujours spécifier le paramètre **QMgrName**, mais sa valeur est ignorée ; les caractères blancs sont une option appropriée.



**Avertissement :** IMS ne parvient pas à se connecter à un groupe de partage de files d'attente.

**Applications IBM MQ MQI client:** pour les applications IBM MQ MQI client, une connexion est tentée pour chaque définition de canal de connexion client avec le nom de gestionnaire de files d'attente spécifié, jusqu'à ce qu'une connexion aboutisse. Toutefois, le gestionnaire de files d'attente doit avoir le même nom que le nom spécifié. Si un nom vide est spécifié, chaque canal de connexion client avec un nom de gestionnaire de files d'attente vide est essayé jusqu'à ce qu'il aboutisse ; dans ce cas, il n'y a pas de vérification par rapport au nom réel du gestionnaire de files d'attente.

Les applications client IBM MQ ne sont pas prises en charge dans z/OS, mais z/OS peut agir en tant que serveur IBM MQ, auquel les applications client IBM MQ peuvent se connecter.

**Groupes de gestionnaires de files d'attente IBM MQ MQI client:** si le nom spécifié commence par un astérisque (\*), le gestionnaire de files d'attente auquel la connexion est établie peut avoir un nom différent de celui spécifié par l'application. Le nom spécifié (sans astérisque) définit un *groupe* de gestionnaires de files d'attente éligibles pour la connexion. L'implémentation en sélectionne une dans le groupe en essayant chacune à son tour jusqu'à ce qu'une connexion soit trouvée. L'ordre dans lequel les connexions sont tentées est influencé par la pondération du canal client et les valeurs d'affinité de connexion des canaux candidats. Si aucun des gestionnaires de files d'attente du groupe n'est disponible pour la connexion, l'appel échoue. Chaque gestionnaire de files d'attente est essayé une seule fois. Si un astérisque seul est spécifié pour le nom, un groupe de gestionnaires de files d'attente par défaut défini par l'implémentation est utilisé.

Les groupes de gestionnaires de files d'attente sont pris en charge uniquement pour les applications s'exécutant dans un environnement MQ-client ; l'appel échoue si une application non client spécifie un nom de gestionnaire de files d'attente commençant par un astérisque. Un groupe est défini en fournissant plusieurs définitions de canal de connexion client avec le même nom de gestionnaire de files d'attente (nom spécifié sans astérisque), pour communiquer avec chacun des gestionnaires de files d'attente du groupe. Le groupe par défaut est défini en fournissant une ou plusieurs définitions de canal de connexion client, chacune avec un nom de gestionnaire de files d'attente vide (la spécification d'un nom entièrement vide a donc le même effet que la spécification d'un astérisque unique pour le nom d'une application client).

Après la connexion à un gestionnaire de files d'attente d'un groupe, une application peut spécifier des blancs de la manière habituelle dans les zones de nom de gestionnaire de files d'attente des descripteurs de message et d'objet pour désigner le nom du gestionnaire de files d'attente auquel l'application s'est connectée (le *gestionnaire de files d'attente local* ). Si l'application doit connaître ce nom, utilisez l'appel MQINQ pour consulter l'attribut de gestionnaire de files d'attente **QMGrName** .

L'ajout d'un astérisque au nom de connexion implique que l'application ne dépend pas de la connexion à un gestionnaire de files d'attente particulier dans le groupe. Les applications appropriées sont les suivantes:

- Applications qui placent des messages mais n'en reçoivent pas.
- Applications qui placent des messages de demande, puis extraient les messages de réponse d'une file d'attente *dynamique temporaire* .

Les applications inadaptées sont celles qui doivent extraire des messages d'une file d'attente particulière sur un gestionnaire de files d'attente particulier ; ces applications ne doivent pas faire précéder le nom d'un astérisque.

Si vous indiquez un astérisque, la longueur maximale du reste du nom est de 47 caractères.

La longueur de ce paramètre est donnée par MQ\_Q\_MGR\_NAME\_LENGTH.

## Hconn

Type: MQHCONN-sortie

Ce descripteur représente la connexion au gestionnaire de files d'attente. Spécifiez-le sur tous les appels de mise en file d'attente de messages ultérieurs émis par l'application. Elle cesse d'être valide lorsque l'appel MQDISC est émis ou lorsque l'unité de traitement qui définit la portée de l'identificateur s'arrête.

IBM MQ fournit désormais la bibliothèque mqm avec des packages client ainsi que des packages serveur. Cela signifie que lorsqu'un appel MQI trouvé dans la bibliothèque mqm est effectué, le type de connexion est vérifié pour voir s'il s'agit d'une connexion client ou serveur, puis l'appel sous-jacent correct est effectué. Par conséquent, un exit auquel est transmis un *Hconn* peut désormais être lié à la bibliothèque mqm, mais utilisé sur une installation client.

*Portée de l'indicateur:* La portée de l'identificateur renvoyé dépend de l'appel utilisé pour se connecter au gestionnaire de files d'attente (MQCONN ou MQCONNX). Si l'appel utilisé est MQCONNX, la portée du descripteur dépend également de l'option MQCNO\_HANDLE\_SHARE\_\* spécifiée dans la zone *Options* de la structure MQCNO.

- Si l'appel est MQCONN ou que l'option MQCNO\_HANDLE\_SHARE\_NONE est spécifiée, le descripteur renvoyé est un descripteur *non partagé* .

La portée d'un descripteur non partagé est la plus petite unité de traitement parallèle prise en charge par la plateforme sur laquelle l'application s'exécute (voir [Tableau 546](#), à la page 695 pour plus de détails) ; le descripteur n'est pas valide en dehors de l'unité de traitement parallèle à partir de laquelle l'appel a été émis.

- Si vous spécifiez l'option MQCNO\_HANDLE\_SHARE\_BLOCK ou MQCNO\_HANDLE\_SHARE\_NO\_BLOCK, le descripteur renvoyé est un descripteur *partagé* .

La portée d'un descripteur partagé est le processus qui possède l'unité d'exécution à partir de laquelle l'appel a été émis ; le descripteur peut être utilisé à partir de n'importe quelle unité

d'exécution appartenant à ce processus. Toutes les plateformes ne prennent pas en charge les unités d'exécution.

- Si l'appel MQCONN ou MQCONNX échoue avec un code achèvement égal à MQCC\_FAILED, la valeur Hconn n'est pas définie.

Tableau 546. Portée des descripteurs non partagés sur différentes plateformes	
Plateforme	Portée du descripteur non partagé
z/OS	<ul style="list-style-type: none"> <li>• CICS: tâche CICS</li> <li>• IMS: la tâche, jusqu'au point de synchronisation suivant (à l'exclusion des sous-tâches de la tâche)</li> <li>• z/OS batch et TSO: la tâche (à l'exclusion des sous-tâches de la tâche)</li> </ul>
IBM i	Travail
AIX and Linux	Unité d'exécution
Applications Windows 32 bits	Unité d'exécution
Applications Windows 64 bits	Unité d'exécution

Sur les applications z/OS for CICS , la valeur renvoyée est:

**MQHC\_DEF\_HCONN**

Descripteur de connexion par défaut.

**CompCode**

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

**MQCC\_OK**

Achèvement réussi.

**MQCC\_WARNING**

Avertissement (achèvement partiel).

**MQCC\_FAILED**

Echec de l'appel.

**Motif**

Type : MQLONG - sortie

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_WARNING:

**MQRC\_ALREADY\_CONNECTED**

(2002, X'7D2') Application déjà connectée.

**MQRC\_CLUSTER\_EXIT\_LOAD\_ERROR**

(2267, X'8DB') Impossible de charger l'exit de charge de travail du cluster.

**MQRC\_SSL\_ALREADY\_INITIALISÉ**

(2391, X' 957') SSL déjà initialisé.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_ADAPTER\_CONN\_LOAD\_ERROR**

(2129, X'851') Impossible de charger le module de connexion de l'adaptateur.

**MQRC\_ADAPTER\_DEFS\_ERREUR**

(2131, X'853') Module de définition de sous-système d'adaptateur incorrect.

**MQRC\_ADAPTER\_DEFS\_LOAD\_ERROR**

(2132, X'854') Impossible de charger le module de définition de sous-système d'adaptateur.

**MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') Adaptateur non disponible.

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

**MQRC\_ADAPTER\_STOCKAGE\_RUPTURE**

(2127, X'84F') Mémoire insuffisante pour l'adaptateur.

**MQRC\_ANOTHER\_Q\_MGR\_CONNECTED**

(2103, X'837') Un autre gestionnaire de files d'attente est déjà connecté.

**MQRC\_API\_EXIT\_ERROR**

(2374, X'946') Echec de l'exit API.

**MQRC\_API\_EXIT\_INIT\_ERROR**

(2375, X' 947') L'initialisation de l'exit API a échoué.

**MQRC\_API\_EXIT\_TERM\_ERREUR**

(2376, X' 948') L'arrêt de l'exit API a échoué.

**MQRC\_ASID\_MISMATCH**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

**MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

**MQRC\_CONN\_ID\_IN\_USE**

(2160, X'870') Identificateur de connexion déjà utilisé.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**MQRC\_CONNECTION\_ERREUR**

(2273, X'8E1') Erreur lors du traitement de l'appel MQCONN.

**MQRC\_CONNECTION\_NOT\_AVAILABLE**

(2568, X'A08') se produit sur un appel MQCONN ou MQCONNX lorsque le gestionnaire de files d'attente ne parvient pas à fournir une connexion du type de connexion demandé sur l'installation en cours. Une connexion client ne peut pas être établie sur une installation serveur uniquement. Une connexion locale ne peut pas être établie sur une installation client uniquement.

**MQRC\_CONNECTION QUIESCING**

(2202, X'89A') Connexion en cours de mise au repos.

**MQRC\_CONNECTION\_STOPPING**

(2203, X'89B') Connexion en cours d'arrêt.

**MQRC\_CRYPTO\_ERREUR\_MATÉRIEL**

(2382, X'94E') Erreur de configuration matérielle cryptographique.

**MQRC\_DUPLICATE\_RECOV\_COORD**

(2163, X'873') Le coordonnateur de reprise existe.

**MQRC\_ENVIRONMENT\_ERROR**

(2012, X'7DC') Appel non valide dans l'environnement.

De plus, sur l'appel MQCONNX, le bloc de contrôle «MQCSP-Paramètres de sécurité», à la page 346 est transmis à partir d'une application CICS ou IMS .

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') Descripteur de connexion non valide.



**MQRC\_HOST\_NOT\_AVAILABLE**

(2538, X'9EA') Un appel MQCONN a été émis par un client pour se connecter à un gestionnaire de files d'attente, mais la tentative d'allocation d'une conversation au système distant a échoué.

**Non-concordance de MQRC\_INSTALLATION\_MISMATCH**

(2583, X'A17') Non concordance entre l'installation du gestionnaire de files d'attente et la bibliothèque sélectionnée.

**MQRC\_KEY\_REPOSITORY\_ERROR**

(2381, X'94D') Référentiel de clés non valide.

**MQRC\_MAX\_CONNS\_LIMIT\_ATTEINTE**

(2025, X'7E9') Nombre maximal de connexions atteint.

**MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') Non autorisé pour l'accès.

**MQRC\_OPEN\_FAILED**

(2137, X'859') L'ouverture de l'objet a échoué.

**MQRC\_Q\_MGR\_NAME\_ERROR**

(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**

(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

**MQRC\_Q\_MGR QUIESCING**

(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') Ressources système disponibles insuffisantes.

**MQRC\_SECURITY\_ERROR (ERREUR DE SECURITE MQ)**

(2063, X'80F') Une erreur de sécurité s'est produite.

**MQRC\_SSL\_INITIALIZATION\_ERROR**

(2393, X' 959') Erreur d'initialisation SSL.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') Mémoire disponible insuffisante.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Remarques sur l'utilisation

1. Le gestionnaire de files d'attente auquel la connexion est établie à l'aide de l'appel MQCONN est appelé *gestionnaire de files d'attente local*.
2. Les files d'attente appartenant au gestionnaire de files d'attente local apparaissent à l'application en tant que files d'attente locales. Il est possible d'insérer et d'extraire des messages de ces files d'attente.

Les files d'attente partagées appartenant au groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local apparaissent dans l'application en tant que files d'attente locales. Il est possible d'insérer et d'extraire des messages de ces files d'attente.

Les files d'attente appartenant à des gestionnaires de files d'attente éloignées apparaissent comme des files d'attente éloignées. Il est possible d'insérer des messages dans ces files d'attente, mais pas d'en extraire.

3. Si le gestionnaire de files d'attente échoue alors qu'une application est en cours d'exécution, l'application doit émettre à nouveau l'appel MQCONN pour obtenir un nouveau descripteur de connexion à utiliser lors des appels IBM MQ suivants. L'application peut émettre l'appel MQCONN périodiquement jusqu'à ce que l'appel aboutisse.

Si une application ne sait pas si elle est connectée au gestionnaire de files d'attente, elle peut émettre un appel MQCONN en toute sécurité pour obtenir un descripteur de connexion. Si l'application est déjà connectée, le descripteur renvoyé est identique à celui renvoyé par l'appel MQCONN précédent, mais avec le code achèvement MQCC\_WARNING et le code anomalie MQRC\_ALREADY\_CONNECTED.

4. Lorsque l'application a fini d'utiliser les appels IBM MQ , elle doit utiliser l'appel MQDISC pour se déconnecter du gestionnaire de files d'attente.
5. Si l'appel MQCONN échoue avec un code achèvement égal à MQCC\_FAILED, la valeur Hconn n'est pas définie.
6. Sous z/OS :
  - Les applications par lots, TSO et IMS doivent émettre l'appel MQCONN pour utiliser les autres appels IBM MQ . Ces applications peuvent se connecter simultanément à plusieurs gestionnaires de files d'attente.

Si le gestionnaire de files d'attente échoue, l'application doit émettre à nouveau l'appel après le redémarrage du gestionnaire de files d'attente pour obtenir un nouveau descripteur de connexion.

Bien que les applications IMS puissent émettre l'appel MQCONN à plusieurs reprises, même lorsqu'elles sont déjà connectées, cela n'est pas recommandé pour les programmes de traitement de messages en ligne (MPP).


- Les applications CICS n'ont pas besoin d'émettre l'appel MQCONN pour utiliser les autres appels IBM MQ , mais elles peuvent le faire si elles le souhaitent ; l'appel MQCONN et l'appel MQDISC sont acceptés. Toutefois, il n'est pas possible de se connecter simultanément à plusieurs gestionnaires de files d'attente.

En cas de défaillance du gestionnaire de files d'attente, ces applications sont automatiquement reconnectées au redémarrage du gestionnaire de files d'attente et n'ont donc pas besoin d'émettre l'appel MQCONN.

7. Sous z/OS, pour définir les gestionnaires de files d'attente disponibles:

- Pour les applications par lots, les programmeurs système peuvent utiliser la macro CSQBDEF pour créer un module (CSQBDEFV) qui définit le nom du gestionnaire de files d'attente par défaut ou le nom du groupe de partage de files d'attente.
- Pour les applications IMS , les programmeurs système peuvent utiliser la macro CSQQDEFX pour créer un module (CSQQDEFV) qui définit les noms des gestionnaires de files d'attente disponibles et spécifie le gestionnaire de files d'attente par défaut.

En outre, chaque gestionnaire de files d'attente doit être défini dans la région de contrôle IMS et dans chaque région dépendante accédant à ce gestionnaire de files d'attente. Pour ce faire, vous devez créer un membre de sous-système dans le IMS. Bibliothèque PROCLIB et identifiez le membre de sous-système dans les régions IMS applicables. Si une application tente de se connecter à un gestionnaire de files d'attente qui n'est pas défini dans le membre de sous-système pour sa région IMS , l'application se termine de manière anormale.

 Pour plus d'informations sur l'utilisation de ces macros, voir [Macros destinés à l'utilisation par le client](#).

8. Sous IBM i, les programmes qui se terminent de manière anormale ne sont pas automatiquement déconnectés du gestionnaire de files d'attente. Ecrivez des applications pour permettre à l'appel MQCONN ou MQCONNX de renvoyer le code achèvement MQCC\_WARNING et le code anomalie MQRC\_ALREADY\_CONNECTED. Utilisez normalement le descripteur de connexion renvoyé dans cette situation.

## Appel C

```
MQCONN (QMgName, &Hconn, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```

MQCHAR48  QMgrName; /* Name of queue manager */
MQHCONN   Hconn;    /* Connection handle */
MQLONG    CompCode; /* Completion code */
MQLONG    Reason;   /* Reason code qualifying CompCode */

```

## Appel COBOL

```
CALL 'MQCONN' USING QMGRNAME, HCONN, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```

** Name of queue manager
01 QMGRNAME PIC X(48).
** Connection handle
01 HCONN     PIC S9(9) BINARY.
** Completion code
01 COMPCODE  PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON    PIC S9(9) BINARY.

```

## Appel PL/I

```
call MQCONN (QMgrName, Hconn, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```

dcl QMgrName char(48); /* Name of queue manager */
dcl Hconn    fixed bin(31); /* Connection handle */
dcl CompCode fixed bin(31); /* Completion code */
dcl Reason   fixed bin(31); /* Reason code qualifying CompCode */

```

## Appel d'assembleur de haut niveau

```
CALL MQCONN,(QMGRNAME,HCONN,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```

QMGRNAME DS CL48 Name of queue manager
HCONN     DS F    Connection handle
COMPCODE  DS F    Completion code
REASON    DS F    Reason code qualifying COMPCODE

```

## Appel Visual Basic

```
MQCONN QMgrName, Hconn, CompCode, Reason
```

Déclarez les paramètres comme suit :

```

Dim QMgrName As String*48 'Name of queue manager'
Dim Hconn    As Long      'Connection handle'
Dim CompCode As Long      'Completion code'
Dim Reason   As Long      'Reason code qualifying CompCode'

```

## MQCONNX - Connexion du gestionnaire de files d'attente (étendue)

L'appel MQCONNX connecte un programme d'application à un gestionnaire de files d'attente. Il fournit un descripteur de connexion de gestionnaire de files d'attente, qui est utilisé par l'application lors des appels IBM MQ suivants.

L'appel MQCONNX est similaire à l'appel MQCONN, sauf que MQCONNX permet de spécifier des options pour contrôler la façon dont l'appel fonctionne.

- Cet appel est pris en charge sur tous les systèmes IBM MQ et les clients IBM MQ connectés à ces systèmes.

Une connexion client ne peut pas être établie sur une installation serveur uniquement et une connexion locale ne peut pas être établie sur une installation client uniquement.

### Syntaxe

MQCONNX (*QMgrName*, *ConnectOpts*, *Hconn*, *CompCode*, *Motif*)

### Paramètres

#### QMgrName

Type: MQCHAR48 -entrée

Pour plus de détails, voir le paramètre **QMgrName** décrit dans [«MQCONN-Connexion du gestionnaire de files d'attente»](#), à la page 692 .

#### ConnectOpts

Type: MQCNO-entrée/sortie

Pour plus d'informations, voir [«MQCNO-Options de connexion»](#), à la page 325.

#### Hconn

Type: MQHCONN-sortie

Ce descripteur représente la connexion au gestionnaire de files d'attente. Spécifiez-le sur tous les appels de mise en file d'attente de messages ultérieurs émis par l'application. Elle cesse d'être valide lorsque l'appel MQDISC est émis ou lorsque l'unité de traitement qui définit la portée de l'identificateur s'arrête.

IBM MQ fournit désormais la bibliothèque mqm avec des packages client ainsi que des packages serveur. Cela signifie que lorsqu'un appel MQI trouvé dans la bibliothèque mqm est effectué, le type de connexion est vérifié pour voir s'il s'agit d'une connexion client ou serveur, puis l'appel sous-jacent correct est effectué. Par conséquent, un exit auquel est transmis un *Hconn* peut désormais être lié à la bibliothèque mqm, mais utilisé sur une installation client.

*Portée de l'indicateur:* La portée de l'identificateur renvoyé dépend de l'appel utilisé pour se connecter au gestionnaire de files d'attente (MQCONN ou MQCONNX). Si l'appel utilisé est MQCONNX, la portée du descripteur dépend également de l'option MQCNO\_HANDLE\_SHARE\_\* spécifiée dans la zone *Options* de la structure MQCNO.

- Si l'appel est MQCONN ou que l'option MQCNO\_HANDLE\_SHARE\_NONE est spécifiée, le descripteur renvoyé est un descripteur *non partagé* .

La portée d'un descripteur non partagé est la plus petite unité de traitement parallèle prise en charge par la plateforme sur laquelle l'application s'exécute (voir [Tableau 547](#), à la page 701 pour plus de détails) ; le descripteur n'est pas valide en dehors de l'unité de traitement parallèle à partir de laquelle l'appel a été émis.

- Si vous spécifiez l'option MQCNO\_HANDLE\_SHARE\_BLOCK ou MQCNO\_HANDLE\_SHARE\_NO\_BLOCK, le descripteur renvoyé est un descripteur *partagé* .

La portée d'un descripteur partagé est le processus qui possède l'unité d'exécution à partir de laquelle l'appel a été émis ; le descripteur peut être utilisé à partir de n'importe quelle unité

d'exécution appartenant à ce processus. Toutes les plateformes ne prennent pas en charge les unités d'exécution.

- Si l'appel MQCONN ou MQCONNX échoue avec un code achèvement égal à MQCC\_FAILED, la valeur Hconn n'est pas définie.

Tableau 547. Portée des descripteurs non partagés sur différentes plateformes	
Plateforme	Portée du descripteur non partagé
z/OS	<ul style="list-style-type: none"> <li>• CICS: tâche CICS</li> <li>• IMS: la tâche, jusqu'au point de synchronisation suivant (à l'exclusion des sous-tâches de la tâche)</li> <li>• z/OS batch et TSO: la tâche (à l'exclusion des sous-tâches de la tâche)</li> </ul>
IBM i	Travail
AIX and Linux	Unité d'exécution
Applications Windows 32 bits	Unité d'exécution
Applications Windows 64 bits	Unité d'exécution

Sur les applications z/OS for CICS , la valeur renvoyée est:

**MQHC\_DEF\_HCONN**

Descripteur de connexion par défaut.

**CompCode**

Type : MQLONG - sortie

Pour plus de détails, voir le paramètre **CompCode** décrit dans [«MQCONN-Connexion du gestionnaire de files d'attente»](#), à la page 692 .

**Motif**

Type : MQLONG - sortie

Les codes suivants peuvent être renvoyés par les appels MQCONN et MQCONNX. Pour obtenir la liste des codes supplémentaires pouvant être renvoyés par l'appel MQCONNX, voir les codes suivants.

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_WARNING:

**MQRC\_ALREADY\_CONNECTED**

(2002, X'7D2') Application déjà connectée.

**MQRC\_CLUSTER\_EXIT\_LOAD\_ERROR**

(2267, X'8DB') Impossible de charger l'exit de charge de travail du cluster.

**MQRC\_SSL\_ALREADY\_INITIALISÉ**

(2391, X' 957') SSL déjà initialisé.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_ADAPTER\_CONN\_LOAD\_ERROR**

(2129, X'851') Impossible de charger le module de connexion de l'adaptateur.

**MQRC\_ADAPTER\_DEFS\_ERREUR**

(2131, X'853') Module de définition de sous-système d'adaptateur incorrect.

**MQRC\_ADAPTER\_DEFS\_LOAD\_ERROR**

(2132, X'854') Impossible de charger le module de définition de sous-système d'adaptateur.

**MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') Adaptateur non disponible.

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

**MQRC\_ADAPTER\_STOCKAGE\_RUPTURE**

(2127, X'84F') Mémoire insuffisante pour l'adaptateur.

**MQRC\_ANOTHER\_Q\_MGR\_CONNECTED**

(2103, X'837') Un autre gestionnaire de files d'attente est déjà connecté.

**MQRC\_API\_EXIT\_ERROR**

(2374, X'946') Echec de l'exit API.

**MQRC\_API\_EXIT\_INIT\_ERROR**

(2375, X' 947') L'initialisation de l'exit API a échoué.

**MQRC\_API\_EXIT\_TERM\_ERREUR**

(2376, X' 948') L'arrêt de l'exit API a échoué.

**MQRC\_ASID\_MISMATCH**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

**MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

**MQRC\_CONN\_ID\_IN\_USE**

(2160, X'870') Identificateur de connexion déjà utilisé.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**MQRC\_CONNECTION\_ERREUR**

(2273, X'8E1') Erreur lors du traitement de l'appel MQCONN.

**MQRC\_CONNECTION\_NOT\_AVAILABLE**

(2568, X'A08') se produit sur un appel MQCONN ou MQCONNX lorsque le gestionnaire de files d'attente ne parvient pas à fournir une connexion du type de connexion demandé sur l'installation en cours. Une connexion client ne peut pas être établie sur une installation serveur uniquement. Une connexion locale ne peut pas être établie sur une installation client uniquement.

**MQRC\_CONNECTION QUIESCING**

(2202, X'89A') Connexion en cours de mise au repos.

**MQRC\_CONNECTION\_STOPPING**

(2203, X'89B') Connexion en cours d'arrêt.

**MQRC\_CRYPTO\_ERREUR\_MATÉRIEL**

(2382, X'94E') Erreur de configuration matérielle cryptographique.

**MQRC\_DUPLICATE\_RECOV\_COORD**

(2163, X'873') Le coordinateur de reprise existe.

**MQRC\_ENVIRONMENT\_ERROR**

(2012, X'7DC') Appel non valide dans l'environnement.

De plus, sur l'appel MQCONNX, le bloc de contrôle «MQCSP-Paramètres de sécurité», à la page 346 est transmis à partir d'une application CICS ou IMS .

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') Descripteur de connexion non valide.

**MQRC\_HOST\_NOT\_AVAILABLE**

(2538, X'9EA') Un appel MQCONN a été émis par un client pour se connecter à un gestionnaire de files d'attente, mais la tentative d'allocation d'une conversation au système distant a échoué.

**Non-concordance de MQRC\_INSTALLATION\_MISMATCH**

(2583, X'A17') Non concordance entre l'installation du gestionnaire de files d'attente et la bibliothèque sélectionnée.

**MQRC\_KEY\_REPOSITORY\_ERROR**

(2381, X'94D') Référentiel de clés non valide.

**MQRC\_MAX\_CONNS\_LIMIT\_ATEINTE**

(2025, X'7E9') Nombre maximal de connexions atteint.

**MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') Non autorisé pour l'accès.

**MQRC\_OPEN\_FAILED**

(2137, X'859') L'ouverture de l'objet a échoué.

**MQRC\_Q\_MGR\_NAME\_ERROR**

(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**

(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

**MQRC\_Q\_MGR QUIESCING**

(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') Ressources système disponibles insuffisantes.

**MQRC\_SECURITY\_ERROR (ERREUR DE SECURITE MQ)**

(2063, X'80F') Une erreur de sécurité s'est produite.

**MQRC\_SSL\_INITIALIZATION\_ERROR**

(2393, X' 959') Erreur d'initialisation SSL.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') Mémoire disponible insuffisante.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') Une erreur inattendue s'est produite.

Les codes anomalie supplémentaires suivants peuvent être renvoyés par l'appel MQCONN:

Si *CompCode* est MQCC\_FAILED:

**ERREUR MQRC\_AIR\_ERROR**

(2385, X' 951') Enregistrement des informations d'authentification non valide.

**MQRC\_AUTH\_INFO\_CONN\_NAME\_ERROR**

(2387, X' 953') Nom de connexion des informations d'authentification incorrect.

**MQRC\_AUTH\_INFO\_REC\_COUNT\_ERROR**

(2383, X'94F') Le nombre d'enregistrements d'informations d'authentification n'est pas valide.

**MQRC\_AUTH\_INFO\_REC\_ERROR**

(2384, X' 950') Zones d'enregistrement d'informations d'authentification non valides.

**MQRC\_AUTH\_INFO\_TYPE\_ERROR**

(2386, X' 952') Type d'informations d'authentification non valide.

**MQRC\_CD\_ERREUR**

(2277, X'8E5') Définition de canal non valide.

**MQRC\_CLIENT\_CONN\_ERROR**

(2278, X'8E6') Zones de connexion client non valides.

**MQRC\_CNO\_ERREUR**

(2139, X'85B') Structure d'options de connexion non valide.

**MQRC\_CONN\_TAG\_IN\_USE**

(2271, X'8DF') Balise de connexion utilisée.

**MQRC\_CONN\_TAG\_NON\_UTILISABLE**

(2350, X'92E') La balise de connexion n'est pas utilisable.

**MQRC\_CSP\_ERREUR**

(2595, X'A23') La structure MQCSP n'est pas valide.

**V 9.3.4****MQRC\_FUNCTION\_NOT\_SUPPORTED**

(2298, X'8FA') La fonction demandée n'est pas disponible dans l'environnement en cours.

**MQRC\_LDAP\_PASSWORD\_ERROR**

(2390, X'956') Mot de passe LDAP non valide.

**MQRC\_LDAP\_USER\_NAME\_ERROR**

(2388, X'954') Zones de nom d'utilisateur LDAP non valides.

**MQRC\_LDAP\_NOM\_UTILISATEUR\_LENGTH\_ERR**

(2389, X'955') Longueur du nom d'utilisateur LDAP incorrecte.

**MQRC\_OPTIONS\_ERROR**

(2046, X'7FE') Options incorrectes ou incohérentes.

**MQRC\_ERREUR DE SCO\_DU**

(2380, X'94C') La structure des options de configuration SSL n'est pas valide.

**MQRC\_SSL\_CONFIG\_ERROR**

(2392, X'958') Erreur de configuration SSL.

**MQRC\_TOKEN\_HORODATAGE\_NOT\_VALID**

(2064, X'810') Le jeton d'authentification n'est pas encore valide ou est arrivé à expiration.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Remarques sur l'utilisation

Pour le langage de programmation Visual Basic, le point suivant s'applique:

- Le paramètre **ConnectOpts** est déclaré comme étant de type MQCNO. Si l'application s'exécute en tant que IBM MQ MQI clientet que vous souhaitez spécifier les paramètres du canal de connexion client, déclarez le paramètre **ConnectOpts** comme étant de type Anyafin que l'application puisse spécifier une structure MQCNOCD lors de l'appel à la place d'une structure MQCNO. Toutefois, cela signifie que le paramètre **ConnectOpts** ne peut pas être vérifié pour s'assurer qu'il s'agit du type de données correct.

## Appel C

```
MQCONN (QMgrName, &ConnectOpts, &Hconn, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQCHAR48 QMgrName;      /* Name of queue manager */
MQCNO    ConnectOpts;   /* Options that control the action of MQCONN */
MQHCONN  Hconn;        /* Connection handle */
MQLONG   CompCode;     /* Completion code */
MQLONG   Reason;       /* Reason code qualifying CompCode */
```

## Appel COBOL

```
CALL 'MQCONN' USING QMGRNAME, CONNECTOPTS, HCONN, COMPCODE,
REASON.
```

Déclarez les paramètres comme suit :

```
** Name of queue manager
01 QMGRNAME PIC X(48).
```



```

** Options that control the action of MQCONN
  01 CONNECTOPTS.
    COPY CMQCNOV.
** Connection handle
  01 HCONN      PIC S9(9) BINARY.
** Completion code
  01 COMPCODE   PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
  01 REASON     PIC S9(9) BINARY.

```

## Appel PL/I

```
call MQCONN (QMgrName, ConnectOpts, Hconn, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```

dcl QMgrName      char(48);      /* Name of queue manager */
dcl ConnectOpts  like MQCNO;    /* Options that control the action of
                                MQCONN */
dcl Hconn        fixed bin(31); /* Connection handle */
dcl CompCode     fixed bin(31); /* Completion code */
dcl Reason       fixed bin(31); /* Reason code qualifying CompCode */

```

## Appel d'assembleur de haut niveau

```
CALL MQCONN, (QMGRNAME,CONNECTOPTS,HCONN,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

QMGRNAME	DS	CL48	Name of queue manager
CONNECTOPTS	CMQCNOA	,	Options that control the action of MQCONN
HCONN	DS	F	Connection handle
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## Appel Visual Basic

```
MQCONN QMgrName, ConnectOpts, Hconn, CompCode, Reason
```

Déclarez les paramètres comme suit :

```

Dim QMgrName      As String*48 'Name of queue manager'
Dim ConnectOpts  As MQCNO      'Options that control the action of'
                                'MQCONN'
Dim Hconn        As Long       'Connection handle'
Dim CompCode     As Long       'Completion code'
Dim Reason       As Long       'Reason code qualifying CompCode'

```

## MQCRTMH-Création d'un descripteur de message

L'appel MQCRTMH renvoie un descripteur de message.

Une application peut utiliser l'appel MQCRTMH sur les appels de mise en file d'attente de messages suivants:

- Utilisez l'appel [MQSETMP](#) pour définir une propriété du descripteur de message.
- Utilisez l'appel [MQINQMP](#) pour demander la valeur d'une propriété du descripteur de message.
- Utilisez l'appel [MQDLTMP](#) pour supprimer une propriété du descripteur de message.

Le descripteur de message peut être utilisé sur les appels MQPUT et MQPUT1 pour associer les propriétés du descripteur de message à celles du message en cours d'insertion. De même, en spécifiant un descripteur de message dans l'appel MQGET, les propriétés du message en cours d'extraction sont accessibles à l'aide du descripteur de message une fois l'appel MQGET terminé.

Utilisez [MQDLTMH](#) pour supprimer le descripteur de message.

## Syntaxe

MQCRTMH (*Hconn*, *CrtMsgHOpts*, *Hmsg*, *CompCode*, *Motif*)

## Paramètres

### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent. Si la connexion au gestionnaire de files d'attente cesse d'être valide et qu'aucun appel IBM MQ n'est en cours d'exécution sur l'identificateur de message, [MQDLTMH](#) est implicitement appelé pour supprimer le message.

Vous pouvez également spécifier la valeur suivante:

#### **MQHC\_UNASSOCIATED\_HCONN**

Le descripteur de connexion ne représente pas une connexion à un gestionnaire de files d'attente particulier.

Lorsque cette valeur est utilisée, l'identificateur de message doit être supprimé avec un appel explicite à [MQDLTMH](#) afin de libérer la mémoire qui lui est allouée ; IBM MQ ne supprime jamais implicitement l'identificateur de message.

Au moins une connexion valide à un gestionnaire de files d'attente doit être établie sur l'unité d'exécution qui crée le descripteur de message, sinon l'appel échoue avec MQRC\_HCONN\_ERROR.

Dans un environnement comportant plusieurs installations sur un seul système, la valeur MQHC\_UNASSOCIATED\_HCONN est limitée à une utilisation avec la première installation chargée dans le processus. Le code anomalie MQRC\_HMSG\_NOT\_AVAILABLE est renvoyé si le descripteur de message est fourni à une installation différente.

Sur les applications z/OS for CICS , l'appel MQCONN peut être omis et vous pouvez spécifier la valeur suivante pour *Hconn*:

#### **CONN MQHC\_DEFAULT**

Descripteur de connexion par défaut

### CrtMsgHOpts

Type: MQCMHO-entrée

Options qui contrôlent l'action de MQCRTMH. Pour plus d'informations, voir [MQCMHO](#) .

### Msg

Type: MQHMSG-sortie

En sortie, un descripteur de message est renvoyé et peut être utilisé pour définir, interroger et supprimer les propriétés du descripteur de message. Initialement, le descripteur de message ne contient aucune propriété.

Un descripteur de message est également associé à un descripteur de message. Initialement, il contient les valeurs par défaut. Les valeurs des zones de descripteur de message associées peuvent être définies et renseignées à l'aide des appels MQSETMP et MQINQMP. L'appel MQDLTMP réinitialise une zone du descripteur de message à sa valeur par défaut.

Si le paramètre *Hconn* est spécifié comme valeur MQHC\_UNASSOCIATED\_HCONN, le descripteur de message renvoyé peut être utilisé sur les appels MQGET, MQPUT ou MQPUT1 avec n'importe

quelle connexion au sein de l'unité de traitement, mais il ne peut être utilisé que par un seul appel IBM MQ à la fois. Si le descripteur est utilisé lorsqu'un deuxième appel IBM MQ tente d'utiliser le même descripteur de message, le deuxième appel IBM MQ échoue avec le code anomalie MQRC\_MSG\_HANDLE\_IN\_USE.

Si le paramètre *Hconn* n'est pas MQHC\_UNASSOCIATED\_HCONN, le descripteur de message renvoyé ne peut être utilisé que sur la connexion spécifiée.

La même valeur de paramètre *Hconn* doit être utilisée sur les appels MQI suivants où ce descripteur de message est utilisé:

- MQDLTMH
- MQSETMP
- MQINQMP
- MQDLTMP
- MQMHBUF
- MQBUFMH

L'identificateur de message renvoyé cesse d'être valide lorsque l'appel MQDLTMH est émis pour l'identificateur de message ou lorsque l'unité de traitement qui définit la portée de l'identificateur s'arrête. MQDLTMH est appelé implicitement si une connexion spécifique est fournie lors de la création du descripteur de message et que la connexion au gestionnaire de files d'attente cesse d'être valide, par exemple, si MQDBC est appelé.

#### **CompCode**

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

##### **MQCC\_OK**

Achèvement réussi.

##### **MQCC\_FAILED**

Echec de l'appel.

#### **Motif**

Type : MQLONG - sortie

Si *CompCode* est MQCC\_OK:

##### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

##### **MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'089C') Carte non disponible.

##### **MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

##### **MQRC\_ASID\_MISMATCH**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

##### **MQRC\_CALL\_IN\_PROGRESS**

(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

##### **ERREUR MQRC\_CMHO\_ERROR**

(2461, X'099D') La structure des options de création de descripteur de message n'est pas valide.

##### **MQRC\_CONNECTION\_BROKEN**

(2273, X'7D9') La connexion au gestionnaire de files d'attente a été perdue.

##### **MQRC\_HANDLE\_NOT\_AVAILABLE**

(2017, X'07E1') Plus de descripteurs disponibles.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') Descripteur de connexion non valide.

**ERREUR MQRC\_HMSG\_ERROR**

(2460, X'099C') Le pointeur de descripteur de message n'est pas valide.

**MQRC\_OPTIONS\_ERROR**

(2046, X'07FE') Options non valides ou non cohérentes.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') Mémoire disponible insuffisante.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

**C**

```
MQCRTMH (Hconn, &CrtMsgHOpts, &Hmsg, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;          /* Connection handle */
MQCMHO   CrtMsgHOpts;   /* Options that control the action of MQCRTMH */
MQHMSG   Hmsg;          /* Message handle */
MQLONG   CompCode;      /* Completion code */
MQLONG   Reason;        /* Reason code qualifying CompCode */
```

**COBOL**

```
CALL 'MQCRTMH' USING HCONN, CRTMSGHOPTS, HMSG, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle
01 HCONN      PIC S9(9) BINARY.
** Options that control the action of MQCRTMH
01 CRTMSGHOPTS.
   COPY CMQCMHOV.
** Message handle
01 HMSG       PIC S9(18) BINARY.
** Completion code
01 COMPCODE   PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON     PIC S9(9) BINARY.
```

**PL/I**

```
call MQCRTMH (Hconn, CrtMsgHOpts, Hmsg, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
dcl Hconn      fixed bin(31); /* Connection handle */
dcl CrtMsgHOpts like MQCMHO; /* Options that control the action of MQCRTMH */
dcl Hmsg       fixed bin(63); /* Message handle */
dcl CompCode   fixed bin(31); /* Completion code */
dcl Reason     fixed bin(31); /* Reason code qualifying CompCode */
```

## High Level Assembler

```
CALL MQCRTMH, (HCONN, CRTMSGHOPTS, HMSG, COMPCODE, REASON)
```

Déclarez les paramètres comme suit :

HCONN	DS	F	Connection handle
CRTMSGHOPTS	CMQCMHOA	,	Options that control the action of MQCRTMH
HMSG	DS	D	Message handle
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## MQCTL-Rappels de contrôle

L'appel MQCTL effectue des actions de contrôle sur les rappels et les descripteurs d'objet ouverts pour une connexion.

### Syntaxe

MQCTL (*Hconn*, *Opération*, *ControlOpts*, *CompCode*, *Motif*)

### Paramètres

#### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS, l'appel MQCONN peut être omis et vous pouvez spécifier la valeur spéciale suivante pour *Hconn*:

#### MQHC\_DEF\_HCONN

Descripteur de connexion par défaut.

#### Opération

Type : MQLONG - entrée

Opération en cours de traitement sur le rappel défini pour le descripteur d'objet spécifié. Vous devez spécifier une seule des options suivantes:

#### DEMARRAGE MQOP

Démarrer la consommation des messages pour toutes les fonctions de consommateur de message définies pour le descripteur de connexion spécifié.

Les rappels s'exécutent sur une unité d'exécution démarrée par le système, qui est différente de toutes les unités d'exécution d'application.

Cette opération permet de contrôler la poignée de connexion fournie au système. Les seuls appels MQI pouvant être émis par une unité d'exécution autre que l'unité d'exécution destinataire sont les suivants:

- MQCTL avec l'opération MQOP\_STOP
- MQCTL avec l'opération MQOP\_SUSPEND
- MQDISC-Exécute MQCTL avec l'opération MQOP\_STOP avant de déconnecter HConn.

MQRC\_HCONN\_ASYNC\_ACTIVE est renvoyé si un appel d'API IBM MQ est émis alors que le descripteur de connexion est démarré et que l'appel ne provient pas d'une fonction de consommateur de message.

Si un consommateur de message arrête la connexion lors de l'appel MQCBCT\_START\_CALL, l'appel MQCTL est renvoyé avec le code anomalie d'échec MQRC\_CONNECTION\_STOPPED.

Cela peut être émis dans une fonction de consommateur. Pour la même connexion que la routine de rappel, son seul objectif est d'annuler une opération MQOP\_STOP précédemment émise.

Cette option n'est pas prise en charge dans les environnements suivants: CICS sous z/OS ou si l'application est liée à une bibliothèque IBM MQ sans unités d'exécution.

### **MQOP\_START\_WAIT**

Démarrer la consommation des messages pour toutes les fonctions de consommateur de message définies pour le descripteur de connexion spécifié.

Les consommateurs de message s'exécutent sur la même unité d'exécution et le contrôle n'est pas renvoyé à l'appelant de MQCTL tant que:

- Publié par l'utilisation des opérations MQCTL MQOP\_STOP ou MQOP\_SUSPEND, ou
- Toutes les routines de consommateur ont été désenregistrées ou suspendues.

Si tous les consommateurs sont désenregistrés ou suspendus, une opération MQOP\_STOP implicite est émise.

Cette option ne peut pas être utilisée à partir d'une routine de rappel, que ce soit pour le descripteur de connexion en cours ou tout autre descripteur de connexion. Si l'appel est tenté, il est renvoyé avec MQRC\_ENVIRONMENT\_ERROR.

Si, à un moment quelconque au cours d'une opération MQOP\_START\_WAIT, il n'y a pas de consommateurs enregistrés et non suspendus, l'appel échoue avec le code anomalie MQRC\_NO\_CALLBACKS\_ACTIVE.

Si, au cours d'une opération MQOP\_START\_WAIT, la connexion est suspendue, l'appel MQCTL renvoie un code anomalie d'avertissement MQRC\_CONNECTION\_SUSPENDED ; la connexion reste démarrée.

L'application peut choisir d'émettre MQOP\_STOP ou MQOP\_RESUME. Dans cette instance, l'opération MQOP\_RESUME se bloque.

Cette option n'est pas prise en charge dans un client à unité d'exécution unique.

### **MQOP\_ARRET**

Arrêtez la consommation des messages et attendez que tous les destinataires aient terminé leurs opérations avant de terminer cette option. Cette opération libère le descripteur de connexion.

Si elle est émise à partir d'une routine de rappel, cette option n'est pas appliquée tant que la routine n'est pas arrêtée. Aucune autre routine de consommateur de message n'est appelée une fois que les routines de consommateur des messages déjà lus sont terminées et après que des appels d'arrêt (si requis) ont été effectués aux routines de rappel.

S'il est émis en dehors d'une routine de rappel, le contrôle ne revient pas à l'appelant tant que les routines de consommateur pour les messages déjà lus ne sont pas terminées et que des appels d'arrêt (si requis) aux rappels n'ont pas été effectués. Les rappels eux-mêmes, cependant, restent enregistrés.

Cette fonction n'a aucun effet sur les messages de lecture anticipée. Vous devez vous assurer que les consommateurs exécutent MQCLOSE (MQCO\_QUIESCE), à partir de la fonction de rappel, pour déterminer s'il existe d'autres messages disponibles à distribuer.

### **MQOP\_SUSPENSION**

Mettez en pause la consommation des messages. Cette opération libère le descripteur de connexion.

Cela n'a aucun effet sur la lecture avant les messages de l'application. Si vous avez l'intention d'arrêter de consommer des messages pendant longtemps, envisagez de fermer la file d'attente et de la rouvrir lorsque la consommation se poursuit.

S'il est émis à partir d'une routine de rappel, il ne prend effet qu'à la fin de la routine. Aucune autre routine de consommateur de message ne sera appelée après la sortie de la routine en cours.

S'il est émis en dehors d'un rappel, le contrôle ne revient pas à l'appelant tant que la routine de consommateur en cours n'est pas terminée et qu'aucun autre n'est appelé.

### **MQOP\_RESUME**

Reprenez la consommation des messages.

Cette option est normalement émise à partir de l'unité d'exécution de l'application principale, mais elle peut également être utilisée à partir d'une routine de rappel pour annuler une demande de suspension antérieure émise dans la même routine.

Si MQOP\_RESUME est utilisé pour reprendre un MQOP\_START\_WAIT, l'opération se bloque.

### **ControlOpts**

Type: MQCTLO-entrée

Options qui contrôlent l'action de MQCTL

Pour plus d'informations sur la structure, voir [MQCTLO](#) .

### **CompCode**

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_WARNING**

Avertissement (achèvement partiel).

#### **MQCC\_FAILED**

Echec de l'appel.

### **Motif**

Type : MQLONG - sortie

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_ADAPTER\_CONV\_LOAD\_ERROR**

(2133, X'855') Impossible de charger les modules de service de conversion de données.

#### **MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') Adaptateur non disponible.

#### **MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

#### **MQRC\_API\_EXIT\_ERROR**

(2374, X'946') Echec de l'exit API.

#### **MQRC\_API\_EXIT\_LOAD\_ERROR**

(2183, X'887') Impossible de charger l'exit API.

#### **MQRC\_ASID\_MISMATCH**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

#### **MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

#### **MQRC\_CALLBACK\_LINK\_ERROR**

(2487, X'9B7') Impossible d'appeler la routine de rappel

#### **MQRC\_CALLBACK\_NON\_ENREGISTREE**

(2448, X' 990') Impossible de désenregistrer, de suspendre ou de reprendre car il n'y a pas de rappel enregistré

**MQRC\_CALLBACK\_ROUTINE\_ERROR**

(2486, X'9B6') Soit CallbackFunction et CallbackName ont été spécifiés sur un appel MQOP\_REGISTER.

Ou bien CallbackFunction ou CallbackName ont été spécifiés mais ne correspondent pas à la fonction de rappel actuellement enregistrée.

**MQRC\_CALLBACK\_TYPE\_ERREUR**

(2483, X'9B3') Zone de type CallBackincorrecte.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

**MQRC\_CBD\_ERREUR**

(2444, X'98C') Le bloc d'options est incorrect.

**MQRC\_CBD\_OPTIONS\_ERREUR**

(2484, X'9B4') Zone d'options MQCBD incorrecte.

**MQRC\_CICS\_WAIT\_FAILED**

(2140, X'85C') Demande d'attente rejetée par CICS.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**MQRC\_CONNECTION\_NOT\_AUTHORIZED**

(2217, X'8A9') Non autorisé pour la connexion.

**MQRC\_CONNECTION QUIESCING**

(2202, X'89A') Connexion en cours de mise au repos.

**MQRC\_CONNECTION\_STOPPING**

(2203, X'89B') Connexion en cours d'arrêt.

**MQRC\_CORREL\_ID\_ERROR**

(2207, X'89F') Erreur d'identificateur de corrélation.

**MQRC\_ENVIRONMENT\_ERROR**

(2012, X'7DC') Appel non valide dans l'environnement.

**MQRC\_FUNCTION\_NOT\_SUPPORTED**

(2298, X'8FA') La fonction demandée n'est pas disponible dans l'environnement en cours.

**MQRC\_GET\_INHIBITED**

(2016, X'7E0') Se bloque pour la file d'attente.

**MQRC\_GLOBAL\_UOW\_CONFLICT**

(2351, X'92F') Conflit entre les unités d'oeuvre globales.

**MQRC\_GMO\_ERROR**

(2186, X'88A') Structure des options de message Get non valide.

**MQRC\_HANDLE\_IN\_USE\_FOR\_UOW**

(2353, X'931') Descripteur en cours d'utilisation pour l'unité d'oeuvre globale.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') Descripteur de connexion non valide.

**MQRC\_HOBJ\_ERROR**

(2019, X'7E3') Descripteur d'objet non valide.

**MQRC\_INCONSISTENT\_BROWSE**

(2259, X'8D3') Spécification de navigation incohérente.

**MQRC\_INCONSISTENT\_UOW**

(2245, X'8C5') Spécification d'unité d'oeuvre incohérente.

**MQRC\_INVALID\_MSG\_UNDER\_CURSOR**

(2246, X'8C6') Message sous le curseur non valide pour la récupération.

**MQRC\_LOCAL\_UOW\_CONFLICT**

(2352, X'930') Conflit entre l'unité d'oeuvre globale et l'unité d'oeuvre locale.



**MQRC\_MATCH\_OPTIONS\_ERROR**

(2247, X'8C7') Options de correspondance non valides.

**MQRC\_MAX\_MSG\_LENGTH\_ERROR**

(2485, X'9B5') Zone de longueur MaxMsg incorrecte

**MQRC\_MD\_ERROR**

(2026, X'7EA') Descripteur de message non valide.

**MQRC\_MODULE\_ENTRY\_NOT\_FOUND**

(2497, X'9C1') Le point d'entrée de fonction spécifié est introuvable dans le module.

**MQRC\_MODULE\_INVALID**

(2496, X'9C0') Le module a été trouvé mais son type est incorrect (32 bit/64 bits) ou il ne s'agit pas d'une dll valide.

**MQRC\_MODULE\_NOT\_FOUND**

(2495, X'9BF') Module introuvable dans le chemin de recherche ou non autorisé à être chargé.

**MQRC\_MSG\_ID\_ERROR**

(2206, X'89E') Erreur d'identificateur de message.

**MQRC\_MSG\_SEQ\_NUMBER\_ERROR**

(2250, X'8CA') Numéro de séquence de message non valide.

**MQRC\_MSG\_TOKEN\_ERROR**

(2331, X'91B') Utilisation du jeton de message non valide.

**MQRC\_NOT\_OPEN\_FOR\_BROWSE**

(2036, X'7F4') File d'attente non ouverte pour exploration.

**MQRC\_NOT\_OPEN\_FOR\_INPUT**

(2037, X'7F5') File d'attente non ouverte pour saisie.

**MQRC\_OBJECT\_CHANGED**

(2041, X'7F9') Définition d'objet modifiée depuis son ouverture.

**MQRC\_OBJECT\_DAMAGED**

(2101, X'835') Objet endommagé.

**ERREUR D'OPERATION\_MQRC**

(2488, X'9B8') Code opération incorrect sur l'appel API

**MQRC\_OPTIONS\_ERROR**

(2046, X'7FE') Options incorrectes ou incohérentes.

**MQRC\_PAGESET\_ERROR**

(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.

**MQRC\_Q\_DELETED**

(2052, X'804') File d'attente supprimée.

**MQRC\_Q\_INDEX\_TYPE\_ERROR**

(2394, X'95A') Type d'index incorrect pour la file d'attente.

**MQRC\_Q\_MGR\_NAME\_ERROR**

(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**

(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

**MQRC\_Q\_MGR QUIESCING**

(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') Ressources système disponibles insuffisantes.

**MQRC\_SIGNAL\_OUTSTANDING**

(2069, X'815') Signal en attente pour ce descripteur.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') Mémoire disponible insuffisante.

**MQRC\_SUPPRESSED\_BY\_EXIT**

(2109, X'83D') Appel supprimé par le programme d'exit.

**MQRC\_SYNCPOINT\_NOT\_AVAILABLE**

(2072, X'818') Prise en charge du point de synchronisation non disponible.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') Une erreur inattendue s'est produite.

**MQRC\_UOW\_ENLISTMENT\_ERROR**

(2354, X'932') Echec de l'inscription dans l'unité d'oeuvre globale.

**MQRC\_UOW\_MIX\_NOT\_SUPPORTED**

(2355, X'933') Mélange d'appels d'unité d'oeuvre non pris en charge.

**MQRC\_UOW\_NOT\_AVAILABLE**

(2255, X'8CF') Unité d'oeuvre non disponible pour le gestionnaire de files d'attente à utiliser.

**MQRC\_WAIT\_INTERVAL\_ERROR**

(2090, X'82A') Intervalle d'attente non valide dans MQGMO.

**MQRC\_WRONG\_GMO\_VERSION**

(2256, X'8D0') Version MQGMO fournie non valide.

**MQRC\_WRONG\_MD\_VERSION**


(2257, X'8D1') Version MQMD fournie non valide.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Remarques sur l'utilisation

1. Les routines de rappel doivent vérifier les réponses de tous les services qu'elles appellent, et si la routine détecte une condition qui ne peut pas être résolue, elle doit émettre une commande MQCB MQOP\_REGISISTER pour empêcher les appels répétés à la routine de rappel.
2. Si vous utilisez la consommation asynchrone dans une application où un gestionnaire de transactions XA gère des transactions globales, y compris des mises à jour de IBM MQ, vous devez prendre en compte les points supplémentaires suivants:
  - a. Il n'est pas valide d'appeler MQCTL (MQOP\_START) pour un **HConn**, une fois qu'il a été créé, après avoir appelé **xa\_open**.

La raison en est que le **HConn** est devenu associé à un contexte XA et qu'il n'est donc pas accessible sur l'unité d'exécution distincte, ou les unités d'exécution, utilisée par le mécanisme de consommation asynchrone.
  - b. Si vous appelez MQCTL (MQOP\_START) dans ce scénario, l'appel échoue avec le code anomalie MQRC\_ASYNC\_XA\_CONFLICT (2350).
  - c. Il est possible d'appeler MQCTL (MQOP\_START\_WAIT) pour un **HConn**, une fois qu'il a été créé, après avoir appelé **xa\_open**.

En effet, cette méthode de démarrage du mécanisme de consommation asynchrone entraîne l'exécution de tous les autres rappels de **HConn** sur l'unité d'exécution où l'appel MQCTL est effectué. Par conséquent, le lien entre le **HConn** et l'unité d'exécution n'est pas perdu.
3.  Sous z/OS, lorsque l'opération est MQOP\_START:
  - Les programmes qui utilisent des routines de rappel asynchrones doivent être autorisés à utiliser z/OS UNIX System Services (z/OS UNIX).
  - Les programmes Language Environment (LE) qui utilisent des routines de rappel asynchrones doivent utiliser l'option d'exécution LE POSIX(ON).
  - Les programmes non LE qui utilisent des routines de rappel asynchrones ne doivent pas utiliser l'interface z/OS UNIX pthread\_create (service de rappel BPX1PTC).

4.  MQCTL n'est pas pris en charge dans l'adaptateur IMS .

**Remarque :** Dans CICS, MQOP\_START n'est pas pris en charge. A la place, utilisez l'appel de fonction MQOP\_START\_WAIT.

## Appel C

```
MQCTL (Hconn, Operation, &ControlOpts, &CompCode, &Reason)
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;          /* Connection handle */
MQLONG   Operation;     /* Operation being processed */
MQCTLO   ControlOpts    /* Options that control the action of MQCTL */
MQLONG   CompCode;      /* Completion code */
MQLONG   Reason;        /* Reason code qualifying CompCode */
```

## Appel COBOL

```
CALL 'MQCTL' USING HCONN, OPERATION, CTLOPTS, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle
01 HCONN PIC S9(9) BINARY.
** Operation
01 OPERATION PIC S9(9) BINARY.
** Control Options
01 CTLOPTS.
   COPY CMQCTLOV.
** Completion code
01 COMPCODE PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON PIC S9(9) BINARY.
```

## Appel PL/I

```
call MQCTL(Hconn, Operation, Ct1Opts, CompCode, Reason)
```

Déclarez les paramètres comme suit :

```
dcl Hconn      fixed bin(31); /* Connection handle */
dcl Operation  fixed bin(31); /* Operation */
dcl Ct1Opts    like MQCTLO;   /* Options that control the action of MQCTL */
dcl CompCode   fixed bin(31); /* Completion code */
dcl Reason     fixed bin(31); /* Reason code qualifying CompCode */
```

## MQDISC-Déconnexion du gestionnaire de files d'attente

L'appel MQDISC interrompt la connexion entre le gestionnaire de files d'attente et le programme d'application et est l'inverse de l'appel MQCONN ou MQCONNX.

- Sous z/OS, toutes les applications qui utilisent la consommation de messages asynchrones, la gestion d'événements ou le rappel, l'unité d'exécution de contrôle principale doit émettre un appel MQDISC avant de s'arrêter. Pour plus d'informations, voir [Consommation asynchrone des messages IBM MQ](#) .
- Sous z/OS, les applications CICS n'ont pas besoin d'émettre cet appel pour se déconnecter du gestionnaire de files d'attente.

Si une application CICS effectue cet appel, il n'a aucun effet à moins qu'un appel MQCONNX antérieur ait été effectué, en spécifiant l'un des éléments suivants:

MQCNO\_SERIALIZE\_CONN\_TAG\_Q\_MGR  
MQCNO\_SERIALIZE\_CONN\_TAG\_QSG  
MQCNO\_RESTRICT\_CONN\_TAG\_Q\_MGR ou  
MQCNO\_RESTRICT\_CONN\_TAG\_QSG

Dans ce cas, tous les descripteurs d'objet actuellement ouverts sont fermés.

## Syntaxe

MQDISC (*Hconn*, *CompCode*, *Raison*)

## Paramètres

### Hconn

Type: MQHCONN-entrée / sortie

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS, vous pouvez omettre l'appel MQCONN et spécifier la valeur suivante pour *Hconn*:

#### **MQHC\_DEF\_HCONN**

Descripteur de connexion par défaut.

Une fois l'appel terminé, le gestionnaire de files d'attente définit *Hconn* sur une valeur qui n'est pas un descripteur valide pour l'environnement. Cette valeur est:

#### **MQHC\_UNUSABLE\_HCONN**

Descripteur de connexion inutilisable.

Sous z/OS, *Hconn* est défini sur une valeur non définie.

### CompCode

Type : MQLONG - sortie

Code achèvement ; il s'agit de l'un des codes suivants:

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_WARNING**

Avertissement (achèvement partiel).

#### **MQCC\_FAILED**

Echec de l'appel.

### Motif

Type : MQLONG - sortie

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_WARNING:

#### **MQRC\_BACKED\_OUT**

(2003, X'7D3') Unité d'oeuvre annulée.

#### **MQRC\_CONN\_TAG\_NOT\_RELÂCHÉE**

(2344, X' 928') La balise de connexion n'a pas été libérée.

#### **MQRC\_OUTCOME\_EN attente**

(2124, X'84C') Le résultat de l'opération de validation est en attente.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_ADAPTER\_DISC\_LOAD\_ERROR**

(2138, X'85A') Impossible de charger le module de déconnexion de l'adaptateur.

**MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') Adaptateur non disponible.

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

**MQRC\_API\_EXIT\_ERROR**

(2374, X'946') Echec de l'exit API.

**MQRC\_API\_EXIT\_INIT\_ERROR**

(2375, X' 947') L'initialisation de l'exit API a échoué.

**MQRC\_API\_EXIT\_TERM\_ERREUR**

(2376, X' 948') L'arrêt de l'exit API a échoué.

**MQRC\_ASID\_MISMATCH**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**MQRC\_CONNECTION\_STOPPING**

(2203, X'89B') Connexion en cours d'arrêt.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') Descripteur de connexion non valide.

**MQRC\_OUTCOME\_MIXTE**

(2123, X'84B') Le résultat de l'opération de validation ou d'exclusion est mélangé.

**MQRC\_PAGESET\_ERROR**

(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.

**MQRC\_Q\_MGR\_NAME\_ERROR**

(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**

(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') Ressources système disponibles insuffisantes.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') Mémoire disponible insuffisante.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Remarques sur l'utilisation

1. Si un appel MQDISC est émis alors que la connexion comporte toujours des objets ouverts sous cette connexion, le gestionnaire de files d'attente ferme ces objets avec les options de fermeture définies sur MQCO\_NONE.
2. Si l'application se termine avec des modifications non validées dans une unité d'oeuvre, la disposition de ces modifications dépend de la façon dont l'application se termine:
  - a. Si l'application émet l'appel MQDISC avant l'arrêt:

- Pour une unité d'oeuvre coordonnée par le gestionnaire de files d'attente, le gestionnaire de files d'attente émet l'appel MQCMIT pour le compte de l'application. L'unité de travail est validée si possible et annulée si ce n'est pas le cas.

- Pour une unité de travail coordonnée en externe, le statut de l'unité de travail ne change pas ; toutefois, le gestionnaire de files d'attente indique généralement que l'unité de travail doit être validée lorsqu'elle est demandée par le coordinateur de l'unité de travail.

Sous z/OS, CICS, IMS (autres que les programmes DL/1 par lots) et les applications RRS sont comme ceci.

- b. Si l'application se termine normalement mais sans émettre l'appel MQDISC, l'action effectuée dépend de l'environnement:

- Sous z/OS, à l'exception des applications MQ Java ou MQ JMS , les actions décrites dans la remarque 2a se produisent.
- Dans tous les autres cas, les actions décrites dans la remarque 2c se produisent.

En raison des différences entre les environnements, assurez-vous que les applications que vous souhaitez porter valident ou réutilisent l'unité de travail avant qu'elles ne se terminent.

- c. Si l'application se termine *de manière anormale* sans émettre l'appel MQDISC, l'unité de travail est annulée.

### 3. Sous z/OS, les points suivants s'appliquent:

- Les applications CICS n'ont pas besoin d'émettre l'appel MQDISC pour se déconnecter du gestionnaire de files d'attente, car le système CICS se connecte lui-même au gestionnaire de files d'attente et l'appel MQDISC n'a aucun effet sur cette connexion.
- CICS, IMS (autres que les programmes par lots DL/1 ) et les applications RRS utilisent des unités de travail qui sont coordonnées par un coordinateur d'unité de travail externe. Par conséquent, l'appel MQDISC n'affecte pas le statut de l'unité de travail (le cas échéant) qui existe lorsque l'appel est émis.

Toutefois, l'appel MQDISC *indique* la fin d'utilisation de la balise de connexion *ConnTag* qui a été associée à la connexion par un appel MQCONNX antérieur émis par l'application. Si une unité d'oeuvre active fait référence à la balise de connexion lors de l'émission de l'appel MQDISC, l'appel se termine avec le code achèvement MQCC\_WARNING et le code anomalie MQRC\_CONN\_TAG\_NOT\_RELÂCHÉE. La balise de connexion ne peut pas être réutilisée tant que le coordinateur de l'unité de travail externe n'a pas résolu l'unité de travail.

**Remarque :** Dans CICS, MQOP\_START n'est pas pris en charge. A la place, utilisez l'appel de fonction MQOP\_START\_WAIT.

## Appel C

```
MQDISC (&Hconn, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;      /* Connection handle */
MQLONG   CompCode;  /* Completion code */
MQLONG   Reason;    /* Reason code qualifying CompCode */
```

## Appel COBOL

```
CALL 'MQDISC' USING HCONN, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```

** Connection handle
01 HCONN      PIC S9(9) BINARY.
** Completion code
01 COMPCODE   PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON     PIC S9(9) BINARY.

```

## Appel PL/I

```
call MQDISC (Hconn, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```

dcl Hconn      fixed bin(31); /* Connection handle */
dcl CompCode   fixed bin(31); /* Completion code */
dcl Reason     fixed bin(31); /* Reason code qualifying CompCode */

```

## Appel de l'assembleur System/390

```
CALL MQDISC,(HCONN,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```

HCONN      DS  F  Connection handle
COMPCODE   DS  F  Completion code
REASON     DS  F  Reason code qualifying COMPCODE

```

## Appel Visual Basic

```
MQDISC Hconn, CompCode, Reason
```

Déclarez les paramètres comme suit :

```

Dim Hconn      As Long 'Connection handle'
Dim CompCode   As Long 'Completion code'
Dim Reason     As Long 'Reason code qualifying CompCode'

```

## MQDLTMH-Suppression du descripteur de message

L'appel MQDLTMH supprime un descripteur de message et est l'inverse de l'appel MQCRTMH.

### Syntaxe

```
MQDLTMH (Hconn, Hmsg, DltMsgHOpts, CompCode, Motif)
```

### Paramètres

#### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente.

La valeur doit correspondre au descripteur de connexion utilisé pour créer le descripteur de message spécifié dans le paramètre **Hmsg**.

Si le descripteur de message a été créé à l'aide de MQHC\_UNASSOCIATED\_HCONN, une connexion valide doit être établie sur l'unité d'exécution qui supprime le descripteur de message, sinon l'appel échoue avec MQRC\_CONNECTION\_BROKEN.

### **Msg**

Type: MQHMSG-entrée/sortie

Il s'agit du descripteur de message à supprimer. La valeur a été renvoyée par un appel MQCRTMH précédent.

Une fois l'appel terminé, le descripteur est défini sur une valeur non valide pour l'environnement. Cette valeur est:

#### **MQHM\_UNUSABLE\_HMSG**

Descripteur de message inutilisable.

Le descripteur de message ne peut pas être supprimé si un autre appel IBM MQ en cours a été transmis avec le même descripteur de message.

### **DltMsgHOpts**

Type: MQDMHO-entrée

Pour plus d'informations, voir [MQDMHO](#) .

### **CompCode**

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

### **Motif**

Type : MQLONG - sortie

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'089C') Carte non disponible.

#### **MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

#### **MQRC\_ASID\_MISMATCH**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

#### **MQRC\_CALL\_IN\_PROGRESS**

(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

#### **MQRC\_CONNECTION\_BROKEN**

(2009, X'07D9') La connexion au gestionnaire de files d'attente a été perdue.

#### **MQRC\_DMHO\_ERREUR**

(2462, X'099E') La structure des options de suppression de descripteur de message n'est pas valide.

#### **ERREUR MQRC\_HMSG\_ERROR**

(2460, X'099C') Le pointeur de descripteur de message n'est pas valide.

#### **MQRC\_MSG\_HANDLE\_IN\_USE**

(2499, X'09C3') Descripteur de message déjà utilisé.



**MQRC\_OPTIONS\_ERROR**

(2046, X'07FE') Options non valides ou non cohérentes.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') Mémoire disponible insuffisante.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

**Appel C**

```
MQDLTMH (Hconn, &Hmsg, &DltMsgHOpts, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;          /* Connection handle */
MQHMSG   Hmsg;          /* Message handle */
MQDMHO   DltMsgHOpts;  /* Options that control the action of MQDLTMH */
MQLONG   CompCode;     /* Completion code */
MQLONG   Reason;       /* Reason code qualifying CompCode */
```

**Appel COBOL**

```
CALL 'MQDLTMH' USING HCONN, HMSG, DLTMSGHOPTS, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle
01  HCONN    PIC S9(9) BINARY.

** Options that control the action of MQDLTMH
01  DLTMSGHOPTS.
COPY CMQDMHOL.

** Completion code
01  COMPCODE PIC S9(9) BINARY.

** Reason code qualifying COMPCODE
01  REASON   PIC S9(9) BINARY.
```

**Appel PL/I**

```
call MQDLTMH (Hconn, Hmsg, DltMsgHOpts, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
dcl Hconn          /* Connection handle */
dcl Hmsg           /* Message handle */
dcl DltMsgHOpts like MQDMHO; /* Options that control the action of MQDLTMH */
dcl CompCode      /* Completion code */
dcl Reason        /* Reason code qualifying CompCode */
```

**Appel d'assembleur de haut niveau**

```
CALL MQDLTMH, (HCONN, HMSG, DLTMSGHOPTS, COMPCODE, REASON)
```

Déclarez les paramètres comme suit :

HCONN	DS	F	Connection handle
HMSG	DS	D	Message handle
DLTMSGHOPTS	CMQDMHOA	,	Options that control the action of MQDLTMH
COMP CODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMP CODE

## MQDLTMP-Propriété de message Delete

L'appel MQDLTMP supprime une propriété d'un descripteur de message et est l'inverse de l'appel MQSETMP.

### Syntaxe

MQDLTMP (*Hconn*, *Hmsg*, *DltPropOpts*, *Nom*, *CompCode*, *Motif*)

### Paramètres

#### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur doit correspondre au descripteur de connexion utilisé pour créer le descripteur de message spécifié dans le paramètre **Hmsg**.

Si le descripteur de message a été créé à l'aide de MQHC\_UNASSOCIATED\_HCONN, une connexion valide doit être établie sur l'unité d'exécution en supprimant le descripteur de message, sinon l'appel échoue avec MQRC\_CONNECTION\_BROKEN.

#### Msg

Type: MQHMSG-entrée

Il s'agit du descripteur de message contenant la propriété à supprimer. La valeur a été renvoyée par un appel MQCRTMH précédent.

#### DltPropOpt

Type: MQDMPO-entrée

Pour plus de détails, voir le type de données [MQDMPO](#).

#### Nom

Type: MQCHARV-entrée

Nom de la propriété à supprimer. Pour plus d'informations sur les noms de propriété, voir [Noms de propriété](#).

Les caractères génériques ne sont pas autorisés dans le nom de la propriété.

#### CompCode

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

##### MQCC\_OK

Achèvement réussi.

##### MQCC\_WARNING

Avertissement (achèvement partiel).

##### MQCC\_FAILED

Echec de l'appel.

#### Motif

Type : MQLONG - sortie

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_WARNING:

**MQRC\_PROPERTY\_NOT\_AVAILABLE**

(2471, X'09A7') Propriété non disponible.

**MQRC\_RFH\_FORMAT\_ERREUR**

(2421, X'0975') Un dossier MQRFH2 contenant des propriétés n'a pas pu être analysé.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'089C') Carte non disponible.

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'0852') Impossible de charger le module de service de l'adaptateur.

**MQRC\_ASID\_MISMATCH**

(2157, X'086D') Les ASID principal et principal diffèrent.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'07D9') La connexion au gestionnaire de files d'attente a été perdue.

**MQRC\_DMPO\_ERREUR**

(2481, X'09B1') La structure des options de suppression des propriétés de message n'est pas valide.

**ERREUR MQRC\_HMSG\_ERROR**

(2460, X'099C') Descripteur de message non valide.

**MQRC\_MSG\_HANDLE\_IN\_USE**

(2499, X'09C3') Descripteur de message déjà utilisé.

**MQRC\_OPTIONS\_ERROR**

(2046, X'07FE') Options non valides ou non cohérentes.

**MQRC\_PROPERTY\_NAME\_ERROR**

(2442, X'098A') Nom de propriété non valide.

**MQRC\_SOURCE\_CCSID\_ERROR**

(2111, X'083F') Identificateur de jeu de caractères codés de nom de propriété non valide.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'0893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Appel C

```
MQDLTMP (Hconn, Hmsg, &DltPropOpts, &Name, &CompCode, &Reason)
```

Déclarez les paramètres comme suit :

```

MQHCONN Hconn;          /* Connection handle */
MQHMSG  Hmsg;          /* Message handle */
MQDMPO  DltPropOpts;  /* Options that control the action of MQDLTMP */
MQCHARV Name;         /* Property name */
MQLONG  CompCode;     /* Completion code */
MQLONG  Reason;       /* Reason code qualifying CompCode */

```

## Appel COBOL

```
CALL 'MQDLTMP' USING HCONN, HMSG, DLTPROPOPTS, NAME, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle
01 HCONN    PIC S9(9) BINARY.
** Message handle
01 HMSG     PIC S9(18) BINARY.
** Options that control the action of MQDLTMP
01 DLTPROPOPTS.
   COPY CMQDMPOV.
** Property name
01 NAME.
   COPY CMQCHRVA.
** Completion code
01 COMPCODE PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON   PIC S9(9) BINARY.
```

## Appel PL/I

```
call MQDLTMP (Hconn, Hmsg, DltPropOpts, Name, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
dcl Hconn      fixed bin(31); /* Connection handle */
dcl Hmsg       fixed bin(63); /* Message handle */
dcl DltPropOpts like MQDMP0; /* Options that control the action of MQDLTMP */
dcl Name       like MQCHARV; /* Property name */
dcl CompCode   fixed bin(31); /* Completion code */
dcl Reason     fixed bin(31); /* Reason code qualifying CompCode */
```

## Appel d'assembleur de haut niveau

```
CALL MQDLTMP,(HCONN,HMSG,DLTPROPOPTS,NAME,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

HCONN	DS	F	Connection handle
HMSG	DS	D	Message handle
DLTPROPOPTS	CMQDMPOA	,	Options that control the action of MQDLTMP
NAME	CMQCHRVA	,	Property name
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## Message d'extraction MQGET

L'appel MQGET récupère un message à partir d'une file d'attente qui a été ouverte à l'aide de l'appel MQOPEN.

### Syntaxe

MQGET (*Hconn, Hobj, MsgDesc, GetMsgOpts, BufferLength, Buffer, DataLength, CompCode, Reason*)

### Paramètres

#### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS , l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour *Hconn*:

### **MQHC\_DEF\_HCONN**

Descripteur de connexion par défaut.

### **HOBJ**

Type : MQHOBJ - entrée

Ce descripteur représente la file d'attente à partir de laquelle un message est récupéré. La valeur de *Hobj* a été renvoyée par un appel MQOPEN précédent. La file d'attente doit être ouverte avec une ou plusieurs des options suivantes (voir «MQOPEN-Objet ouvert», à la page 766) :

- MQOO\_INPUT\_SHARED
- MQOO\_INPUT\_EXCLUSIVE
- MQOO\_INPUT\_AS\_Q\_DEF
- MQOO\_BROWSE

### **MsgDesc**

Type : MQMD - entrée/sortie

Cette structure décrit les attributs du message requis ainsi que ceux du message récupéré. Pour plus d'informations, voir «MQMD-Descripteur de message», à la page 438.

Si *BufferLength* est inférieur à la longueur du message, *MsgDesc* est rempli par le gestionnaire de files d'attente, si MQGMO\_ACCEPT\_TRUNCATED\_MSG est spécifié dans le paramètre **GetMsgOpts** (voir [Zone MQGMO-Options](#) ).

Si l'application fournit un MQMD version-1 , le message renvoyé a un MQMDE préfixé aux données du message d'application, mais uniquement si une ou plusieurs des zones du MQMDE ont une valeur autre que la valeur par défaut. Si toutes les zones de MQMDE possèdent des valeurs par défaut, MQMDE est omis. Un nom de format MQFMT\_MD\_EXTENSION dans la zone *Format* de MQMD indique qu'une structure MQMDE est présente.

L'application n'a pas besoin de fournir une structure MQMD si un descripteur de message valide est fourni dans la zone *MsgHandle*. Si cette zone n'est pas renseignée, le descripteur de message est extrait du descripteur associé aux descripteurs de messages.

Si l'application fournit un descripteur de message au lieu d'une structure MQMD et spécifie l'option MQGMO\_PROPERTIES\_FORCE\_MQRFH2, l'appel échoue avec le code anomalie MQRC\_MD\_ERROR. L'appel échoue également, avec le code anomalie MQRC\_MD\_ERROR, si l'application ne fournit pas de structure MQMD et spécifie MQGMO\_PROPERTIES\_AS\_Q\_DEF, et que l'attribut de file d'attente **PropertyControl** est MQPROP\_FORCE\_MQRFH2.

Si des options de concordance sont spécifiées et que le descripteur de message associé est utilisé, les zones de saisie utilisées pour la correspondance proviennent du descripteur de message.

### **GetMsgOpts**

Type : MQGMO - entrée/sortie

Pour plus d'informations, voir «Options MQGMO-Get-message», à la page 382.

### **BufferLength**

Type : MQLONG - entrée

Il s'agit de la longueur en octets de la zone *Buffer* . Indiquez 0 pour les messages qui ne comportent aucune donnée ou si le message doit être retiré de la file d'attente et les données supprimées (vous devez spécifier MQGMO\_ACCEPT\_TRUNCATED\_MSG dans ce cas).

**Remarque :** La longueur du message le plus long qu'il est possible de lire à partir de la file d'attente est donnée par l'attribut de file d'attente **MaxMsgLength** ; voir «Attributs des files d'attente», à la page 875.

## Mémoire tampon

Type : MQBYTEExBufferLength - sortie

Il s'agit de la zone devant contenir les données de message. Alignez la mémoire tampon sur une limite appropriée à la nature des données dans le message. L'alignement à 4 octets convient à la plupart des messages (y compris les messages contenant des structures d'en-tête IBM MQ), mais certains messages peuvent nécessiter un alignement plus strict. Par exemple, un message comportant un entier binaire de 64 bits peut avoir besoin d'un alignement sur 8 octets.

Si *BufferLength* est inférieur à la longueur du message, la plus grande partie possible du message est déplacée dans **Buffer**. Cela se produit si MQGMO\_ACCEPT\_TRUNCATED\_MSG est spécifié sur le paramètre **GetMsgOpts** (voir [Zone MQGMO-Options](#) pour plus d'informations).

Le jeu de caractères et le codage des données dans **Buffer** sont fournis par les zones *CodedCharSetId* et *Encoding* renvoyées dans le paramètre **MsgDesc**. Si ces valeurs sont différentes des valeurs requises par le récepteur, ce dernier doit convertir les données de message d'application dans le jeu de caractères et le codage requis. L'option MQGMO\_CONVERT peut être utilisée (avec un exit écrit par l'utilisateur, si nécessaire) pour convertir les données de message ; voir «Options MQGMO-Get-message», à la page 382 pour en savoir plus sur cette option.

**Remarque :** Tous les autres paramètres de l'appel MQGET se trouvent dans le jeu de caractères et le codage du gestionnaire de files d'attente local (indiqués par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et MQENC\_NATIVE).

Si l'appel échoue, le contenu de la mémoire tampon peut avoir changé.

En langage de programmation C, le paramètre est déclaré en tant que pointeur nul : vous pouvez spécifier l'adresse de n'importe quel type de données en tant que paramètre.

Si la valeur du paramètre **BufferLength** est zéro, il n'est pas fait référence à *Buffer* ; dans ce cas, l'adresse de paramètre transmise par les programmes écrits en assembleur C ou System/390 peut être null.

## DataLength

Type : MQLONG - sortie

Il s'agit de la longueur en octets des données d'application *dans le message*. Si cette valeur est supérieure à *BufferLength*, seuls *BufferLength* octets sont renvoyés dans le paramètre **Buffer** (c'est-à-dire que le message est tronqué). Si la valeur est zéro, le message ne contient aucune donnée d'application.

Si *BufferLength* est inférieur à la longueur du message, *DataLength* est toujours exécuté par le gestionnaire de files d'attente, si MQGMO\_ACCEPT\_TRUNCATED\_MSG est spécifié dans le paramètre **GetMsgOpts** (voir [Zone MQGMO-Options](#) pour plus d'informations). L'application peut ainsi choisir la taille de la mémoire tampon requise pour contenir les données de message, puis exécuter de nouveau l'appel avec une taille de mémoire tampon appropriée.

Toutefois, si l'option MQGMO\_CONVERT est spécifiée et que les données de message converties sont trop longues pour tenir dans *Buffer*, la valeur renvoyée pour *DataLength* est la suivante :

- Longueur des données *non converties*, pour les formats définis par le gestionnaire de files d'attente.

Dans ce cas, si la nature des données provoque son extension lors de la conversion, l'application doit allouer une mémoire tampon supérieure à la valeur renvoyée par le gestionnaire de files d'attente pour *DataLength*.

- la valeur renvoyée par l'exit de conversion de données pour les formats d'application.

## CompCode

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

### MQCC\_OK

Achèvement réussi.

**MQCC\_WARNING**

Avertissement (achèvement partiel).

**MQCC\_FAILED**

Echec de l'appel.

**Motif**

Type : MQLONG - sortie

Les codes raison répertoriés sont ceux que le gestionnaire de files d'attente peut renvoyer pour le paramètre **Reason** . Si l'application spécifie l'option MQGMO\_CONVERT et qu'un exit écrit par l'utilisateur est appelé pour convertir une partie ou la totalité des données de message, l'exit détermine la valeur renvoyée pour le paramètre **Reason** . Par conséquent, les valeurs autres que celles décrites sont possibles.

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_WARNING:

**MQRC\_CONVERTED\_MSG\_TOO\_BIG**

(2120, X'848') Données converties trop volumineuses pour la mémoire tampon.

**MQRC\_CONVERTED\_STRING\_TOO\_BIG**

(2190, X'88E') Chaîne convertie trop volumineuse pour la zone.

**MQRC\_DBCS\_ERROR**

(2150, X'866') Chaîne DBCS non valide.

**MQRC\_FORMAT\_ERROR**

(2110, X'83E') Format de message non valide.

**MQRC\_INCOMPLETE\_GROUP**

(2241, X'8C1') Groupe de messages non complet.

**MQRC\_INCOMPLETE\_MSG**

(2242, X'8C2') Message logique non complet.

**MQRC\_INCONSISTENT\_CCSDS**

(2243, X'8C3') Les segments de message possèdent des CCSID différents.

**MQRC\_INCONSISTENT\_ENCODINGS**

(2244, X'8C4') Les segments de message possèdent des codages différents.

**MQRC\_INCONSISTENT\_UOW**

(2245, X'8C5') Spécification d'unité d'oeuvre incohérente.

**MQRC\_MSG\_TOKEN\_ERROR**

(2331, X'91B') Utilisation de jeton de message non valide.

**MQRC\_NO\_MSG\_LOCKED**

(2209, X'8A1') Aucun message verrouillé.

**MQRC\_NOT\_CONVERTED**

(2119, X'847') Données de message non converties.

**MQRC\_OPTIONS\_CHANGED**

(nnnn, X'xxx') Les options devant être cohérentes ont été modifiées.

**MQRC\_PARTIALLY\_CONVERTED**

(2272, X'8E0') Données de message partiellement converties.

**MQRC\_SIGNAL\_REQUEST\_ACCEPTED**

(2070, X'816') Aucun message renvoyé (mais la demande de signal a été acceptée).

**MQRC\_SOURCE\_BUFFER\_ERROR**

(2145, X'861') Paramètre de tampon source non valide.

**MQRC\_SOURCE\_CCSDS\_ERROR**

(2111, X'83F') Identificateur de jeu de caractères codés source non valide.

**MQRC\_SOURCE\_DECIMAL\_ENC\_ERROR**

(2113, X'841') Codage décimal condensé dans le message non reconnu.

**MQRC\_SOURCE\_FLOAT\_ENC\_ERROR**

(2114, X'842') Codage à virgule flottante dans le message non reconnu.

**MQRC\_SOURCE\_INTEGER\_ENC\_ERROR**

(2112, X'840') Codage d'entier source non reconnu.

**MQRC\_SOURCE\_LENGTH\_ERROR**

(2143, X'85F') Paramètre de longueur source non valide.

**MQRC\_TARGET\_BUFFER\_ERROR**

(2146, X'862') Paramètre de tampon cible non valide.

**MQRC\_TARGET\_CCSDID\_ERROR**

(2115, X'843') Identificateur de jeu de caractères codés cible non valide.

**MQRC\_TARGET\_DECIMAL\_ENC\_ERROR**

(2117, X'845') Codage décimal condensé spécifié par le récepteur non reconnu.

**MQRC\_TARGET\_FLOAT\_ENC\_ERROR**

(2118, X'846') Codage à virgule flottante spécifié par le récepteur non reconnu.

**MQRC\_TARGET\_INTEGER\_ENC\_ERROR**

(2116, X'844') Codage d'entier cible non reconnu.

**MQRC\_TRUNCATED\_MSG\_ACCEPTED**

(2079, X'81F') Message tronqué renvoyé (traitement terminé).

**MQRC\_TRUNCATED\_MSG\_FAILED**

(2080, X'820') Message tronqué renvoyé (traitement non terminé).

Si *CompCode* est MQCC\_FAILED:

**MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') Adaptateur non disponible.

**MQRC\_ADAPTER\_CONV\_LOAD\_ERROR**

(2133, X'855') Impossible de charger les modules de service de conversion de données.

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

**MQRC\_API\_EXIT\_ERROR**

(2374, X'946') Echec de l'exit API.

**MQRC\_API\_EXIT\_LOAD\_ERROR**

(2183, X'887') Impossible de charger l'exit API.

**MQRC\_ASID\_MISMATCH**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

**MQRC\_BACKED\_OUT**

(2003, X'7D3') Unité d'oeuvre annulée.

**MQRC\_BUFFER\_ERROR**

(2004, X'7D4') Paramètre de tampon non valide.

**MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

**MQRC\_CF\_NOT\_XX\_ENCODE\_CASE\_ONE disponible**

(2345, X' 929') Unité de couplage non disponible.

**MQRC\_CF\_STRUC\_FAILED**

(2373, X'945') Echec de la structure de l'unité de couplage.

**MQRC\_CF\_STRUC\_IN\_USE**

(2346, X'92A') Structure de l'unité de couplage en cours d'utilisation.



**MQRC\_CF\_STRUC\_LIST\_HDR\_IN\_USE**  
(2347, X'92B') En-tête de liste de la structure d'unité de couplage en cours d'utilisation.

**MQRC\_CICS\_WAIT\_FAILED**  
(2140, X'85C') Demande d'attente rejetée par CICS.

**MQRC\_CONNECTION\_BROKEN**  
(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**MQRC\_CONNECTION\_NOT\_AUTHORIZED**  
(2217, X'8A9') Non autorisé pour la connexion.

**MQRC\_CONNECTION QUIESCING**  
(2202, X'89A') Connexion en cours de mise au repos.

**MQRC\_CONNECTION\_STOPPING**  
(2203, X'89B') Connexion en cours d'arrêt.

**MQRC\_CORREL\_ID\_ERROR**  
(2207, X'89F') Erreur d'identificateur de corrélation.

**MQRC\_DATA\_LENGTH\_ERROR**  
(2010, X'7DA') Paramètre de longueur des données non valide.

**MQRC\_DB2\_NOT\_AVAILABLE**  
(2342, X'926') Sous-système Db2 non disponible.

**MQRC\_GET\_INHIBITED**  
(2016, X'7E0') Se bloque pour la file d'attente.

**MQRC\_GLOBAL\_UOW\_CONFLICT**  
(2351, X'92F') Conflit entre les unités d'oeuvre globales.

**MQRC\_GMO\_ERROR**  
(2186, X'88A') Structure des options de message Get non valide.

**MQRC\_HANDLE\_IN\_USE\_FOR\_UOW**  
(2353, X'931') Descripteur en cours d'utilisation pour l'unité d'oeuvre globale.

**MQRC\_HCONN\_ERROR**  
(2018, X'7E2') Descripteur de connexion non valide.

**MQRC\_HOBJ\_ERROR**  
(2019, X'7E3') Descripteur d'objet non valide.

**MQRC\_INCONSISTENT\_BROWSE**  
(2259, X'8D3') Spécification de navigation incohérente.

**MQRC\_INCONSISTENT\_UOW**  
(2245, X'8C5') Spécification d'unité d'oeuvre incohérente.

**MQRC\_INVALID\_MSG\_UNDER\_CURSOR**  
(2246, X'8C6') Message sous le curseur non valide pour la récupération.

**MQRC\_LOCAL\_UOW\_CONFLICT**  
(2352, X'930') Conflit entre l'unité d'oeuvre globale et l'unité d'oeuvre locale.

**MQRC\_MATCH\_OPTIONS\_ERROR**  
(2247, X'8C7') Options de correspondance non valides.

**MQRC\_MD\_ERROR**  
(2026, X'7EA') Descripteur de message non valide.

**MQRC\_MSG\_ID\_ERROR**  
(2206, X'89E') Erreur d'identificateur de message.

**MQRC\_MSG\_SEQ\_NUMBER\_ERROR**  
(2250, X'8CA') Numéro de séquence de message non valide.

**MQRC\_MSG\_TOKEN\_ERROR**  
(2331, X'91B') Utilisation du jeton de message non valide.

**MQRC\_NO\_MSG\_AVAILABLE**  
(2033, X'7F1') Aucun message disponible.

**MQRC\_NO\_MSG\_UNDER\_CURSOR**  
(2034, X'7F2') Curseur d'exploration non positionné sur le message.

**MQRC\_NOT\_OPEN\_FOR\_BROWSE**  
(2036, X'7F4') File d'attente non ouverte pour exploration.

**MQRC\_NOT\_OPEN\_FOR\_INPUT**  
(2037, X'7F5') File d'attente non ouverte pour saisie.

**MQRC\_OBJECT\_CHANGED**  
(2041, X'7F9') Définition d'objet modifiée depuis son ouverture.

**MQRC\_OBJECT\_DAMAGED**  
(2101, X'835') Objet endommagé.

**MQRC\_OPTIONS\_ERROR**  
(2046, X'7FE') Options incorrectes ou incohérentes.

**MQRC\_PAGESET\_ERROR**  
(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.

**MQRC\_Q\_DELETED**  
(2052, X'804') File d'attente supprimée.

**MQRC\_Q\_INDEX\_TYPE\_ERROR**  
(2394, X'95A') Type d'index incorrect pour la file d'attente.

**MQRC\_Q\_MGR\_NAME\_ERROR**  
(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**  
(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

**MQRC\_Q\_MGR QUIESCING**  
(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.

**MQRC\_Q\_MGR\_STOPPING**  
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**MQRC\_RESOURCE\_PROBLEM**  
(2102, X'836') Ressources système disponibles insuffisantes.

**MQRC\_SECOND\_MARK\_NOT\_ALLOWED**  
(2062, X'80E') Un message est déjà marqué.

**MQRC\_SIGNAL\_OUTSTANDING**  
(2069, X'815') Signal en attente pour ce descripteur.

**MQRC\_SIGNAL1\_ERROR**  
(2099, X'833') Zone de signal non valide.

**MQRC\_STORAGE\_MEDIUM\_FULL**  
(2192, X'890') Support de stockage externe saturé.

**MQRC\_STORAGE\_NOT\_AVAILABLE**  
(2071, X'817') Mémoire disponible insuffisante.

**MQRC\_SUPPRESSED\_BY\_EXIT**  
(2109, X'83D') Appel supprimé par le programme d'exit.

**MQRC\_SYNCPOINT\_LIMIT\_REACHED**  
(2024, X'7E8') Aucun autre message pouvant être traité au sein de l'unité d'oeuvre en cours.

**MQRC\_SYNCPOINT\_NOT\_AVAILABLE**  
(2072, X'818') Prise en charge du point de synchronisation non disponible.

**MQRC\_UNEXPECTED\_ERROR**  
(2195, X'893') Une erreur inattendue s'est produite.

**MQRC\_UOW\_ENLISTMENT\_ERROR**  
(2354, X'932') Echec de l'inscription dans l'unité d'oeuvre globale.

**MQRC\_UOW\_MIX\_NOT\_SUPPORTED**  
(2355, X'933') Mélange d'appels d'unité d'oeuvre non pris en charge.

**MQRC\_UOW\_NOT\_AVAILABLE**

(2255, X'8CF') Unité d'oeuvre non disponible pour le gestionnaire de files d'attente à utiliser.

**MQRC\_WAIT\_INTERVAL\_ERROR**

(2090, X'82A') Intervalle d'attente non valide dans MQGMO.

**MQRC\_WRONG\_GMO\_VERSION**

(2256, X'8D0') Version MQGMO fournie non valide.

**MQRC\_WRONG\_MD\_VERSION**

(2257, X'8D1') Version MQMD fournie non valide.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Remarques sur l'utilisation

1. Le message extrait est normalement supprimé de la file d'attente. Cette suppression peut avoir lieu dans le cadre de l'appel MQGET proprement dit ou d'un point de synchronisation.

Les options d'exploration sont les suivantes : MQGMO\_BROWSE\_FIRST, MQGMO\_BROWSE\_NEXT et MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR.

2. Si l'option MQGMO\_LOCK est spécifiée à l'aide d'une des options d'exploration, le message exploré est verrouillé de sorte qu'il ne soit visible que par ce descripteur.

Si l'option MQGMO\_UNLOCK est spécifiée, un message précédemment verrouillé est déverrouillé. Aucun message n'est extrait dans ce cas, et les paramètres **MsgDesc**, **BufferLength**, **Bufferet** et **DataLength** ne sont pas vérifiés ou modifiés.

3. Pour les applications émettant un appel MQGET, le message extrait peut être perdu si l'application se termine de façon anormale ou si la connexion est interrompue pendant le traitement de l'appel. Ce problème se produit car le substitut s'exécutant sur la même plateforme que le gestionnaire de files d'attente qui émet l'appel MQGET pour le compte de l'application ne peut pas détecter la perte de l'application tant que le substitut n'est pas sur le point de renvoyer le message à l'application, une fois que le message a été supprimé de la file d'attente. Cela peut se produire tant pour les messages persistants que pour les messages non persistants.

Pour éliminer le risque de perte de messages de cette façon, extrayez systématiquement les messages dans des unités d'oeuvre. C'est-à-dire, en spécifiant l'option MQGMO\_SYNCPOINT sur l'appel MQGET et en utilisant l'appel MQCMIT ou MQBACK pour valider ou annuler l'unité d'oeuvre une fois le traitement des messages terminé. Si l'option MQGMO\_SYNCPOINT est spécifiée et que le client prend fin de façon anormale ou que la connexion est interrompue, le substitut annule l'unité d'oeuvre sur le gestionnaire de files d'attente et le message est rétabli dans la file d'attente. Pour plus d'informations sur les points de synchronisation, voir [Considérations relatives aux points de synchronisation dans les applications IBM MQ](#).

Cette situation peut se produire avec les clients IBM MQ ainsi qu'avec les applications qui s'exécutent sur la même plateforme que le gestionnaire de files d'attente.

4. Si une application place une séquence de messages dans une file d'attente particulière au sein d'une unité d'oeuvre unique et qu'elle valide ensuite cette unité d'oeuvre correctement, les messages deviennent disponibles pour extraction, comme suit :
  - Si la file d'attente est une *file d'attente non partagée* (c'est-à-dire une file d'attente locale), tous les messages de l'unité d'oeuvre deviennent disponibles en même temps.
  - Si la file d'attente est une *file d'attente partagée*, les messages de l'unité d'oeuvre sont disponibles dans l'ordre dans lequel ils ont été insérés, mais pas tous en même temps. Lorsque le système est extrêmement sollicité, il est possible que le premier message de l'unité d'oeuvre soit extrait correctement mais que l'appel MQGET correspondant au deuxième message ou à un message ultérieur dans l'unité d'oeuvre échoue en raison du code MQRC\_NO\_MSG\_AVAILABLE. Si cela se produit, l'application doit attendre un petit moment avant de relancer l'opération.
5. Si une application place une séquence de messages dans la même file d'attente sans utiliser de groupes de messages, l'ordre de ces messages est préservé sous réserve que certaines conditions soient satisfaites. Pour plus de détails, voir [Remarques sur l'utilisation de MQPUT](#). Si les conditions

sont satisfaites, les messages sont présentés à l'application réceptrice dans l'ordre dans lequel ils ont été envoyés, sous les réserves suivantes :

- Un seul récepteur extrait des messages de la file d'attente.

Si deux applications ou plus extraient des messages de la file d'attente, elles doivent convenir avec l'émetteur du mécanisme à utiliser pour identifier les messages faisant partie d'une séquence. Par exemple, l'expéditeur peut affecter à toutes les zones `CorrelId` des messages d'une séquence une valeur unique à cette séquence de messages.

- Le récepteur ne modifie pas délibérément l'ordre d'extraction, par exemple en spécifiant un `MsgId` ou un `CorrelId` particulier.

Si l'application émettrice place les messages sous la forme d'un groupe de messages, les messages sont présentés à l'application réceptrice dans l'ordre correct sous réserve que l'application réceptrice spécifie l'option `MQGMO_LOGICAL_ORDER` dans l'appel `MQGET`. Pour plus d'informations sur les groupes de messages, voir :

- [MQMD - Zone MsgFlags](#)
- [MQPMO\\_LOGICAL\\_ORDER](#)
- [MQGMO\\_LOGICAL\\_ORDER](#)

Si l'utilisateur extrait des messages d'un groupe dans le cadre d'un point de synchronisation, il doit s'assurer que le groupe entier est traité avant de tenter de terminer la transaction.

6. Les applications doivent tester le code retour `MQFB_QUIT` dans la zone `Feedback` du paramètre **MsgDesc** et se terminer si elles trouvent cette valeur. Pour plus d'informations, voir [MQMD - Zone Feedback](#).
7. Si la file d'attente identifiée par `Hobj` a été ouverte avec l'option `MQOO_SAVE_ALL_CONTEXT` et que le code achèvement de l'appel `MQGET` est `MQCC_OK` ou `MQCC_WARNING`, le contexte associé au descripteur de file d'attente `Hobj` est défini sur le contexte du message qui a été extrait (sauf si l'option `MQGMO_BROWSE_FIRST`, `MQGMO_BROWSE_NEXT` ou `MQGMO_BROWSE_MSG_UNDER_CURSOR` est définie, auquel cas le contexte est marqué comme non disponible).

Vous pouvez utiliser le contexte enregistré d'un appel `MQPUT` ou `MQPUT1` ultérieur en spécifiant les options `MQPMO_PASS_IDENTITY_CONTEXT` ou `MQPMO_PASS_ALL_CONTEXT`. Cela permet de transférer en intégralité ou en partie le contexte du message reçu vers un autre message (par exemple, lorsque le message est réacheminé vers une autre file d'attente). Pour plus d'informations sur le contexte de message, voir [Contexte de message](#).

8. Si vous incluez l'option `MQGMO_CONVERT` dans le paramètre **GetMsgOpts**, les données de message d'application sont converties en la représentation demandée par l'application de réception, avant que les données ne soient placées dans le paramètre **Buffer** :
  - La zone `Format` dans les informations de contrôle du message identifie la structure des données d'application et les zones `CodedCharSetId` et `Encoding` dans les informations de contrôle du message spécifient son identificateur de jeu de caractères et son codage.
  - L'application émettant l'appel `MQGET` spécifie dans les zones `CodedCharSetId` et `Encoding` du paramètre **MsgDesc** l'identificateur de jeu de caractères et le codage dans lesquels convertir les données de message d'application.

Lorsque la conversion des données de message est nécessaire, elle est effectuée soit par le gestionnaire de files d'attente lui-même, soit par un exit écrit par l'utilisateur, en fonction de la valeur de la zone `Format` dans les informations de contrôle du message:

- Les noms de format suivants correspondent aux formats convertis par le gestionnaire de files d'attente ; ces formats sont appelés formats "intégrés" :
  - `MQFMT_ADMIN`
  - `MQFMT_CICS` ( z/OS uniquement)
  - `MQFMT_COMMAND_1`

- MQFMT\_COMMAND\_2
  - MQFMT\_DEAD\_LETTER\_HEADER
  - MQFMT\_DIST\_HEADER
  - MQFMT\_EVENT version 1
  - MQFMT\_EVENT version 2 ( z/OS uniquement)
  - MQFMT\_IMS
  - MQFMT\_IMS\_VAR\_STRING
  - MQFMT\_MD\_EXTENSION
  - MQFMT\_PCF
  - MQFMT\_REF\_MSG\_HEADER
  - MQFMT\_RF\_HEADER
  - MQFMT\_RF\_HEADER\_2
  - MQFMT\_STRING
  - MQFMT\_TRIGGER
  - MQFMT\_WORK\_INFO\_HEADER ( z/OS uniquement)
  - MQFMT\_XMIT\_Q\_HEADER
- Le nom de format MQFMT\_NONE est une valeur spéciale indiquant que la nature des données du message n'est pas définie. En conséquence, le gestionnaire de files d'attente ne tente pas la conversion lorsque le message est extrait de la file d'attente.

**Remarque :** Si MQGMO\_CONVERT est spécifié dans l'appel MQGET pour un message dont le nom de format est MQFMT\_NONE et que le jeu de caractères ou le codage du message est différent de celui spécifié dans le paramètre **MsgDesc** , le message est renvoyé dans le paramètre **Buffer** (en supposant qu'aucune autre erreur ne se soit produite), mais l'appel se termine avec le code achèvement MQCC\_WARNING et le code anomalie MQRC\_FORMAT\_ERROR.

Vous pouvez utiliser MQFMT\_NONE lorsque la nature des données de message signifie qu'une conversion n'est pas nécessaire ou lorsque les applications émettrice et réceptrice sont convenus du format d'envoi des données de message.

- Tous les autres noms de format transmettent le message à un exit écrit par l'utilisateur, à des fins de conversion. L'exit porte le même nom que le format, hormis des ajouts spécifiques à l'environnement. Les noms de format spécifiés par l'utilisateur ne doivent pas commencer par les lettres IBM MQ.

Pour plus de détails sur l'exit de conversion de données, voir [«Exit de conversion de données»](#), à la page 951.

Les données utilisateur du message peuvent être converties d'un jeu de caractères pris en charge à l'autre et d'un codage à l'autre. Toutefois, sachez que si le message contient une ou plusieurs structures d'en-tête IBM MQ , il ne peut pas être converti depuis ou vers un jeu de caractères comportant des caractères codés sur deux octets ou des caractères codés sur plusieurs octets pour l'un des caractères admis dans les noms de file d'attente. Cette tentative donne lieu au code anomalie MQRC\_SOURCE\_CCSD\_ERROR ou MQRC\_TARGET\_CCSD\_ERROR et le message est renvoyé non converti. Le jeu de caractères Unicode UTF-16 est un exemple de ce type de jeu de caractères.

Renvoyé par MQGET, le code anomalie suivant indique que la conversion du message a abouti :

- MQRC\_NONE

Le code anomalie suivant indique que le message a peut-être été converti ; l'application doit vérifier les zones CodedCharSetId et Encoding du paramètre **MsgDesc** pour le savoir:

- MQRC\_TRUNCATED\_MSG\_ACCEPTED

Tous les autres codes anomalie indiquent que le message n'a pas été converti.

**Remarque** : L'interprétation de ce code anomalie est vraie pour les conversions effectuées par un exit écrit par l'utilisateur uniquement si l'exit est conforme aux instructions de traitement décrites dans «Exit de conversion de données», à la page 951.

9. Lors de l'utilisation de l'interface orientée objet pour l'extraction des messages, vous pouvez choisir de ne pas indiquer de mémoire tampon pour le stockage des données de message pour un appel MQGET. Lorsque vous obtenez un message à l'aide d'une application orientée objet sans restreindre la taille de la mémoire tampon de réception des messages, l'application n'échoue pas avec MQRC\_CONVERTED\_MSG\_TOO\_BIG et reçoit le message converti. Cela concerne les environnements suivants :

- .NET, y compris les applications entièrement gérées
- C++
- Java ( IBM MQ classes for Java )

**Remarque** : Pour tous les clients, si la valeur de `sharingConversations` est zéro et si la mémoire tampon est trop petite pour recevoir le message converti, le message non converti est renvoyé avec le code anomalie MQRC\_CONVERTED\_MSG\_TOO\_BIG. Pour plus d'informations sur `sharingConversations`, voir [Utilisation du partage de conversations dans une application client](#).

10. Pour les formats intégrés, le gestionnaire de files d'attente peut exécuter la *conversion par défaut* des chaînes de caractères du message lorsque l'option MQGMO\_CONVERT est spécifiée. La conversion par défaut permet au gestionnaire de files d'attente d'utiliser, lors de la conversion des données de type chaîne, un jeu de caractères par défaut spécifié par l'installation qui se rapproche du jeu de caractères réel. En conséquence, l'appel MQGET peut aboutir avec le code achèvement MQCC\_OK, au lieu de se terminer avec MQCC\_WARNING et le code anomalie MQRC\_SOURCE\_CCSD\_ERROR ou MQRC\_TARGET\_CCSD\_ERROR.

**Remarque** : L'utilisation d'un jeu de caractères approximatif pour la conversion des données de type chaîne donne lieu à la conversion éventuellement incorrecte de certains caractères. Pour éviter cela, utilisez des caractères dans la chaîne qui sont communs au jeu de caractères réel et au jeu de caractères par défaut.

La conversion par défaut s'applique aux données de message d'application et aux zones alphanumériques des structures MQMD et MQMDE :

- La conversion par défaut des données de message d'application se produit uniquement lorsque toutes les instructions suivantes sont vraies:
  - L'application spécifie MQGMO\_CONVERT.
  - Le message contient des données qui doivent être converties à partir ou vers un jeu de caractères non pris en charge.
  - La conversion par défaut a été activée lors de l'installation ou du redémarrage du gestionnaire de files d'attente.
- La conversion par défaut des zones alphanumériques dans les structures MQMD et MQMDE a lieu selon les besoins, si la conversion par défaut est activée pour le gestionnaire de files d'attente. La conversion est exécutée même si l'option MQGMO\_CONVERT n'est pas spécifiée par l'application lors de l'appel MQGET.

11. Pour le langage de programmation Visual Basic, les points suivants s'appliquent :

- Si la taille du paramètre **Buffer** est inférieure à la longueur spécifiée par le paramètre **BufferLength**, l'appel échoue avec le code anomalie MQRC\_STORAGE\_NOT\_AVAILABLE.
- Le paramètre **Buffer** est déclaré comme étant de type `String`. Si les données à extraire de la file d'attente ne sont pas de type `String`, utilisez l'appel MQGETAny à la place de MQGET.

L'appel MQGETAny possède les mêmes paramètres que l'appel MQGET, sauf que le paramètre **Buffer** est déclaré comme étant de type `Any`, ce qui permet d'extraire n'importe quel type de données. Toutefois, cela signifie que `Buffer` ne peut pas être vérifié pour s'assurer qu'il a une taille d'au moins `BufferLength` octets.

12. Les options MQGET ne sont pas toutes prises en charge lorsque la lecture anticipée est activée. Le tableau ci-dessous présente les options admises et indique si elles peuvent être modifiées d'un appel MQGET à l'autre.

Tableau 548. Options MQGET admises lorsque la lecture anticipée est activée

	Admise lorsque la lecture anticipée est activée, et modifiable d'un appel MQGET à l'autre	Autorisé lorsque la lecture anticipée est activée mais ne peut pas être modifiée entre les appels MQGET <sup>a</sup>	Les options MQGET qui ne sont pas autorisées lorsque la lecture anticipée est activée <sup>b</sup>
Valeurs MQGET MD	MsgId <sup>c</sup> CorrelId <sup>c</sup>	Codage CodedCharSetId	
Options MQGET MQGMO	MQGMO_WAIT MQGMO_NO_WAIT MQGMO_FAIL_IF QUIESCING MQGMO_BROWSE_FIRST <sup>d</sup> MQGMO_BROWSE_NEXT <sup>d</sup> MQGMO_BROWSE_MESSAGE _UNDER_CURSOR <sup>d</sup>	MQGMO_SYNCPOINT_IF_PERSISTENT MQGMO_NO_SYNCPOINT MQGMO_ACCEPT_TRUNCATED_MSG MQGMO_CONVERT MQGMO_LOGICAL_ORDER MQGMO_COMPLETE_MSG MQGMO_ALL_MSGS_AVAILABLE MQGMO_ALL_SEGMENTS_AVAILABLE MQGMO_MARK_BROWSE_HANDLE MQGMO_MARK_BROWSE_CO_OP MQGMO_UNMARK_BROWSE_CO_OP MQGMO_UNMARK_BROWSE_HANDLE MQGMO_UNMARKED_BROWSE_MSG MQGMO_PROPERTIES_FORCE_MQRFH2 MQGMO_NO_PROPERTIES MQGMO_PROPERTIES_IN_HANDLE MQGMO_PROPERTIES_COMPATIBILITY	MQGMO_SET_SIGNAL MQGMO_SYNCPOINT MQGMO_MARK_SKIP _BACKOUT MQGMO_MSG_UNDER _CURSOR <sup>d</sup> MQGMO_LOCK MQGMO_UNLOCK
Valeurs MQGMO		MsgHandle	

- Si ces options sont modifiées d'un appel MQGET à l'autre, un code anomalie MQRC\_OPTIONS\_CHANGED est renvoyé.
  - Si ces options sont spécifiées lors du premier appel MQGET, la lecture anticipée est désactivée. Si ces options sont spécifiées lors d'un appel MQGET ultérieur, un code anomalie MQRC\_OPTIONS\_ERROR est renvoyé.
  - Les applications client doivent être informées que, si les valeurs MsgId et CorrelId sont modifiées d'un appel MQGET à l'autre, il se peut que les messages comportant les valeurs précédentes aient déjà été envoyés au client et qu'ils restent dans la mémoire tampon de lecture anticipée du client tant qu'ils ne sont pas consommés (ou automatiquement purgés).
  - Le premier appel MQGET détermine si les messages doivent être explorés ou extraits d'une file d'attente lorsque la lecture anticipée est activée. Si l'application tente d'utiliser une combinaison d'exploration et d'extraction, un code anomalie MQRC\_OPTIONS\_CHANGED est renvoyé.
  - MQGMO\_MSG\_UNDER\_CURSOR n'est pas possible avec la lecture anticipée. Les messages peuvent être explorés ou extraits lorsque la lecture anticipée est activée mais pas une combinaison des deux.
13. Les applications peuvent extraire de façon destructive des messages non validés seulement si ces messages sont placés dans la même unité d'oeuvre locale que l'unité d'oeuvre d'extraction. Les applications ne peuvent pas extraire de messages de façon non destructive.
14. Les messages indiqués par un curseur d'exploration peuvent être extraits dans une unité d'oeuvre. Cette méthode ne permet pas d'extraire un message non validé.

## Appel C

```
MQGET (Hconn, Hobj, &MsgDesc, &GetMsgOpts, BufferLength, Buffer,
      &DataLength, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;          /* Connection handle */
MQHOBJ   Hobj;           /* Object handle */
MQMD     MsgDesc;        /* Message descriptor */
MQGMO    GetMsgOpts;     /* Options that control the action of MQGET */
MQLONG   BufferLength;    /* Length in bytes of the Buffer area */
MQBYTE   Buffer[n];      /* Area to contain the message data */
MQLONG   DataLength;     /* Length of the message */
MQLONG   CompCode;      /* Completion code */
MQLONG   Reason;        /* Reason code qualifying CompCode */
```

## Appel COBOL

```
CALL 'MQGET' USING HCONN, HOBJ, MSGDESC, GETMSGOPTS, BUFFERLENGTH,  
BUFFER, DATALENGTH, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle  
01 HCONN          PIC S9(9) BINARY.  
** Object handle  
01 HOBJ          PIC S9(9) BINARY.  
** Message descriptor  
01 MSGDESC.  
   COPY CMQMDV.  
** Options that control the action of MQGET  
01 GETMSGOPTS.  
   COPY CMQGMV.  
** Length in bytes of the BUFFER area  
01 BUFFERLENGTH PIC S9(9) BINARY.  
** Area to contain the message data  
01 BUFFER        PIC X(n).  
** Length of the message  
01 DATALENGTH  PIC S9(9) BINARY.  
** Completion code  
01 COMPCODE     PIC S9(9) BINARY.  
** Reason code qualifying COMPCODE  
01 REASON       PIC S9(9) BINARY.
```

## Appel PL/I

```
call MQGET (Hconn, Hobj, MsgDesc, GetMsgOpts, BufferLength, Buffer,  
DataLength, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
dcl Hconn          fixed bin(31); /* Connection handle */  
dcl Hobj          fixed bin(31); /* Object handle */  
dcl MsgDesc       like MQMD;    /* Message descriptor */  
dcl GetMsgOpts    like MQGMO;   /* Options that control the action of  
                                MQGET */  
dcl BufferLength   fixed bin(31); /* Length in bytes of the Buffer  
                                area */  
dcl Buffer         char(n);      /* Area to contain the message data */  
dcl DataLength    fixed bin(31); /* Length of the message */  
dcl CompCode      fixed bin(31); /* Completion code */  
dcl Reason        fixed bin(31); /* Reason code qualifying CompCode */
```

## Appel d'assembleur de haut niveau

```
CALL MQGET, (HCONN,HOBJ,MSGDESC,GETMSGOPTS,BUFFERLENGTH,  
BUFFER,DATALENGTH,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

HCONN	DS	F	Connection handle
HOBJ	DS	F	Object handle
MSGDESC	CMQMDA	,	Message descriptor
GETMSGOPTS	CMQGMOA	,	Options that control the action of MQGET
BUFFERLENGTH	DS	F	Length in bytes of the BUFFER area
BUFFER	DS	CL(n)	Area to contain the message data
DATALENGTH	DS	F	Length of the message
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE



## Appel Visual Basic

```
MQGET Hconn, Hobj, MsgDesc, GetMsgOpts, BufferLength, Buffer,  
DataLength, CompCode, Reason
```

Déclarez les paramètres comme suit :

```
Dim Hconn      As Long  'Connection handle'  
Dim Hobj       As Long  'Object handle'  
Dim MsgDesc    As MQMD  'Message descriptor'  
Dim GetMsgOpts As MQGMO 'Options that control the action of MQGET'  
Dim BufferLength As Long 'Length in bytes of the Buffer area'  
Dim Buffer      As String 'Area to contain the message data'  
Dim DataLength As Long  'Length of the message'  
Dim CompCode   As Long  'Completion code'  
Dim Reason     As Long  'Reason code qualifying CompCode'
```

## MQINQ-Attributs d'objet Inquire

L'appel MQINQ renvoie un tableau d'entiers et un ensemble de chaînes de caractères contenant les attributs d'un objet.

Les types d'objet suivants sont valides:

- Gestionnaire de files d'attente
- File d'attente
- Liste de noms
- Définition de processus

## Syntaxe

MQINQ (*Hconn, Hobj, SelectorCount, Sélecteurs, IntAttrCount, IntAttrs, CharAttrLength, CharAttrs, CompCode, Motif*)

## Paramètres

### Hconn

Type: MQHCONN -entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS , l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour *Hconn*:

### MQHC\_DEF\_HCONN

Descripteur de connexion par défaut.

### HOBJ

Type: MQHOBJ -entrée

Ce descripteur représente l'objet (de tout type) avec les attributs requis. Le descripteur doit être renvoyé par un appel MQOPEN précédent qui a spécifié l'option MQOO\_INQUIRE .

### SelectorCount

Type: MQLONG -entrée

Nombre de sélecteurs fournis dans le tableau *Selectors* . Il s'agit du nombre d'attributs à renvoyer. Zéro est une valeur valide. Le nombre maximal autorisé est 256.

### Sélecteurs

Type: MQLONG x *SelectorCount* -entrée

Il s'agit d'un tableau de sélecteurs d'attribut **SelectorCount** ; chaque sélecteur identifie un attribut (entier ou caractère) avec une valeur obligatoire.

Chaque sélecteur doit être valide pour le type d'objet que *Hobj* représente, sinon l'appel échoue avec le code achèvement MQCC\_FAILED et le code anomalie MQRC\_SELECTOR\_ERROR.

Dans le cas particulier des files d'attente:

- Si le sélecteur n'est pas valide pour les files d'attente de n'importe quel type, l'appel échoue avec le code achèvement MQCC\_FAILED et le code anomalie MQRC\_SELECTOR\_ERROR.
- Si le sélecteur s'applique uniquement aux files d'attente de types autres que le type de l'objet, l'appel aboutit avec le code achèvement MQCC\_WARNING et le code anomalie MQRC\_SELECTOR\_NOT\_FOR\_TYPE.
- Si la file d'attente en cours d'interrogation est une file d'attente de cluster, les sélecteurs valides dépendent de la manière dont la file d'attente a été résolue ; voir «Remarques sur l'utilisation», à la page 754 pour plus de détails.

Vous pouvez spécifier des sélecteurs dans n'importe quel ordre. Les valeurs d'attribut qui correspondent à des sélecteurs d'attribut de type entier (sélecteurs MQIA\_\*) sont renvoyées dans *IntAttrs* dans l'ordre dans lequel ces sélecteurs apparaissent dans *Selectors*. Les valeurs d'attribut qui correspondent à des sélecteurs d'attribut de caractères (sélecteurs MQCA\_\*) sont renvoyées dans *CharAttrs* dans l'ordre dans lequel ces sélecteurs se produisent. Les sélecteurs MQIA\_\* peuvent être imbriqués avec les sélecteurs MQCA\_\* ; seul l'ordre relatif au sein de chaque type est important.

#### Remarque :

1. Les sélecteurs d'attribut de type entier et caractère sont alloués dans deux plages différentes ; les sélecteurs MQIA\_\* résident dans la plage MQIA\_FIRST à MQIA\_LAST et les sélecteurs MQCA\_\* dans la plage MQCA\_FIRST à MQCA\_LAST.  
  
Pour chaque plage, les constantes MQIA\_LAST\_USED et MQCA\_LAST\_USED définissent la valeur la plus élevée acceptée par le gestionnaire de files d'attente.
2. Si tous les sélecteurs MQIA\_\* apparaissent en premier, les mêmes numéros d'élément peuvent être utilisés pour adresser les éléments correspondants dans les tableaux *Selectors* et *IntAttrs*.
3. Si le paramètre **SelectorCount** a pour valeur zéro, il n'est pas fait référence à *Selectors*. Dans ce cas, l'adresse de paramètre transmise par les programmes écrits en assembleur C ou S/390 peut être null.

Les attributs que vous pouvez consulter sont répertoriés dans les tableaux suivants. Pour les sélecteurs MQCA\_\*, la constante qui définit la longueur en octets de la chaîne résultante dans *CharAttrs* est fournie entre parenthèses.

Les tableaux qui suivent répertorient les sélecteurs, par objet, dans l'ordre alphabétique, comme suit:

- Sélecteurs d'attribut [Tableau 549](#), à la page 739 MQINQ pour les files d'attente
- Sélecteurs d'attribut [Tableau 550](#), à la page 741 MQINQ pour les listes de noms
- Sélecteurs d'attribut [Tableau 551](#), à la page 742 MQINQ pour les définitions de processus
- Sélecteurs d'attribut [Tableau 552](#), à la page 742 MQINQ pour le gestionnaire de files d'attente

Tous les sélecteurs sont pris en charge sur toutes les plateformes IBM MQ, sauf indication contraire dans la colonne **Remarque** comme suit:

#### Non z/OS

Pris en charge sur toutes les plateformes **sauf** z/OS

#### z/OS

Pris en charge **uniquement** sur z/OS

Tableau 549. Sélecteurs d'attribut MQINQ pour les files d'attente

Sélecteur	Longueur de la zone	Description	Remarque
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	Date de la dernière modification	
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	Heure de la dernière modification	
MQCA_BACKOUT_REQ_Q_NAME	MQ_Q_NAME_LENGTH	Nombre excessif de noms de remise en file d'attente d'annulation	
MQCA_BASE_Q_NAME	MQ_Q_NAME_LENGTH	Nom de la file d'attente dans laquelle l'alias est résolu	
MQCA_CF_STRUC_NAME	MQ_CF_STRUC_NAME_LENGTH	Nom de la structure de l'unité de couplage	z/OS
MQCA_CLUS_CHL_NAME	MQ_CHANNEL_NAME_LENGTH	Nom du canal émetteur de cluster qui utilise cette file d'attente comme file d'attente de transmission.	
MQCA_CLUSTER_NAME	MQ_CLUSTER_NAME_LENGTH	Nom du cluster	
MQCA_CLUSTER_NAMELIST	MQ_NAMELIST_NAME_LENGTH	Liste de noms de cluster	
MQCA_CREATION_DATE	MQ_CREATION_DATE_LENGTH	Date de création de file d'attente	
MQCA_CREATION_TIME	MQ_CREATION_TIME_LENGTH	Heure de création de file d'attente	
MQCA_CUSTOM	MQ_CUSTOM_LENGTH	Attribut personnalisé pour les nouvelles fonctions	
MQCA_INITIATION_Q_NAME	MQ_Q_NAME_LENGTH	Nom de la file d'attente d'initialisation	
MQCA_PROCESS_NAME	MQ_PROCESS_NAME_LENGTH	Nom de la définition de processus	
MQCA_Q_DESC	MQ_Q_DESC_LENGTH	Description de file d'attente	
MQCA_Q_NAME	MQ_Q_NAME_LENGTH	Nom de la file d'attente	
MQCA_REMOTE_Q_MGR_NAME	MQ_Q_MGR_NAME_LENGTH	Nom du gestionnaire de files d'attente éloignées	
MQCA_REMOTE_Q_NAME	MQ_Q_NAME_LENGTH	Nom de la file d'attente éloignée connue sur le gestionnaire de files d'attente éloignées	
MQCA_STORAGE_CLASS	MQ_STORAGE_CLASS_LENGTH	Nom de la classe de stockage	z/OS
MQCA_TRIGGER_DATA	MQ_TRIGGER_DATA_LENGTH	Données de déclenchement	
MQCA_XMIT_Q_NAME	MQ_Q_NAME_LENGTH	Nom de la file d'attente de transmission	
MQIA_ACCOUNTING_Q	MQLONG	Contrôle la collecte des données comptables pour la file d'attente	Non z/OS
MQIA_BACKOUT_THRESHOLD	MQLONG	Seuil d'annulation	

Tableau 549. Sélecteurs d'attribut MQINQ pour les files d'attente (suite)

Sélecteur	Longueur de la zone	Description	Remarque
MQIA_CLWL_Q_PRIORITY	MQLONG	Priorité de la file d'attente	
MQIA_CLWL_Q_RANK	MQLONG	Rang de la file d'attente	
MQIA_CLWL_USEQ	MQLONG	Utilisation des files d'attente éloignées	
MQIA_CURRENT_Q_DEPTH	MQLONG	Nombre de messages dans la file d'attente	
MQIA_DEF_BIND	MQLONG	Liaison par défaut	
MQIA_DEF_INPUT_OPEN_OPTION	MQLONG	Option open-for-input par défaut	
MQIA_DEF_PERSISTENCE	MQLONG	Persistance par défaut	
MQIA_DEF_PRIORITY	MQLONG	Priorité par défaut	
MQIA_DEFINITION_TYPE	MQLONG	Type de définition de file d'attente	
MQIA_DIST_LISTS	MQLONG	Prise en charge de la liste de diffusion	Non z/OS
MQIA_HARDEN_GET_BACKOUT	MQLONG	Indique si le nombre d'annulations doit être durci	
MQIA_INDEX_TYPE	MQLONG	Type d'index géré pour la file d'attente	z/OS
MQIA_INHIBIT_GET	MQLONG	Indique si les opérations d'extraction sont autorisées	
MQIA_INHIBIT_PUT	MQLONG	Indique si les opérations d'insertion sont autorisées	
MQIA_MAX_MSG_LENGTH	MQLONG	Longueur maximale des messages	
MQIA_MAX_Q_DEPTH	MQLONG	Nombre maximal de messages autorisés dans la file d'attente	
MQIA_MSG_DELIVERY_SEQUENCE	MQLONG	Indique si la priorité du message est pertinente	
MQIA_NPM_CLASS	MQLONG	Niveau de fiabilité des messages non persistants	
MQIA_OPEN_INPUT_COUNT	MQLONG	Nombre d'appels MQOPEN ayant la file d'attente ouverte pour l'entrée	
MQIA_OPEN_OUTPUT_COUNT	MQLONG	Nombre d'appels MQOPEN dont la file d'attente est ouverte pour la sortie	
MQIA_PROPERTY_CONTROL	MQLONG	Attribut de contrôle de propriété	
MQIA_Q_DEPTH_HIGH_EVENT	MQLONG	Attribut de contrôle pour les événements de longueur élevée de file d'attente	Non z/OS
MQIA_Q_DEPTH_HIGH_LIMIT	MQLONG	Limite supérieure de la longueur de la file d'attente	Non z/OS

<i>Tableau 549. Sélecteurs d'attribut MQINQ pour les files d'attente (suite)</i>			
<b>Sélecteur</b>	<b>Longueur de la zone</b>	<b>Description</b>	<b>Remarque</b>
MQIA_Q_DEPTH_LOW_EVENT	MQLONG	Attribut de contrôle pour les événements de longueur faible de la file d'attente	Non z/OS
MQIA_Q_DEPTH_LOW_LIMIT	MQLONG	Limite inférieure de la longueur de la file d'attente	Non z/OS
MQIA_Q_DEPTH_MAX_EVENT	MQLONG	Attribut de contrôle du nombre maximal d'événements de longueur de file d'attente	Non z/OS
MQIA_Q_SERVICE_INTERVAL	MQLONG	Limite pour l'intervalle de service de file d'attente	Non z/OS
MQIA_Q_SERVICE_INTERVAL_EVENT	MQLONG	Attribut de contrôle pour les événements d'intervalle de service de file d'attente	Non z/OS
MQIA_Q_TYPE	MQLONG	Type de file d'attente	
MQIA_QSG_DISP	MQLONG	Disposition d'un groupe de partage de files d'attente	z/OS
MQIA_RETENTION_INTERVAL	MQLONG	Intervalle de conservation des files d'attente	
MQIA_SCOPE	MQLONG	Portée de définition de file d'attente	Non z/OS
MQIA_SHAREABILITY	MQLONG	Indique si la file d'attente peut être partagée pour l'entrée	
MQIA_STATISTICS_Q	MQLONG	Contrôle la collecte des données statistiques pour la file d'attente	Non z/OS
MQIA_TRIGGER_CONTROL	MQLONG	Contrôle du déclenchement	
MQIA_TRIGGER_DEPTH	MQLONG	Longueur de déclenchement	
MQIA_TRIGGER_MSG_PRIORITY	MQLONG	Priorité des messages de seuil des déclencheurs	
MQIA_TRIGGER_TYPE	MQLONG	Type de déclenchement	
MQIA_USAGE	MQLONG	Utilisation	

<i>Tableau 550. Sélecteurs d'attribut MQINQ pour les listes de noms</i>			
<b>Sélecteur</b>	<b>Longueur de la zone</b>	<b>Description</b>	<b>Remarque</b>
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	Date de la modification la plus récente	
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	Heure de l'altération la plus récente	
MQCA_NAMELIST_DESC	MQ_NAMELIST_DESC_LENGTH	Description de la liste de noms	
MQCA_NAMELIST_NAME	MQ_NAMELIST_NAME_LENGTH	Nom de l'objet liste de noms	

Tableau 550. Sélecteurs d'attribut MQINQ pour les listes de noms (suite)

Sélecteur	Longueur de la zone	Description	Remarque
MQIA_NAMELIST_TYPE	MQLONG	Type de liste de noms	z/OS
MQCA_NAMES	MQ_Q_NAME_LENGTH <i>x Number of names in the list</i>	Noms dans la liste de noms	
MQIA_NAME_COUNT	MQLONG	Nombre de noms dans la liste de noms	
MQIA_QSG_DISP	MQLONG	Disposition d'un groupe de partage de files d'attente	z/OS

Tableau 551. Sélecteurs d'attribut MQINQ pour les définitions de processus

Sélecteur	Longueur de la zone	Description	Remarque
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	Date de la modification la plus récente	
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	Heure de l'altération la plus récente	
MQCA_APPL_ID	MQ_PROCESS_APPL_ID_LENGTH	Identificateur d'application	
MQCA_ENV_DATA	MQ_PROCESS_ENV_DATA_LENGTH	Données d'environnement	
MQCA_PROCESS_DESC	MQ_PROCESS_DESC_LENGTH	Description de la définition de processus	
MQCA_PROCESS_NAME	MQ_PROCESS_NAME_LENGTH	Nom de la définition de processus	
MQCA_USER_DATA	MQ_PROCESS_USER_DATA_LENGTH	Données utilisateur	
MQIA_APPL_TYPE	MQLONG	Type d'application	
MQIA_QSG_DISP	MQLONG	Disposition d'un groupe de partage de files d'attente	z/OS

Tableau 552. Sélecteurs d'attribut MQINQ pour le gestionnaire de files d'attente

Sélecteur	Longueur de la zone	Description	Remarque
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	Date de la modification la plus récente	
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	Heure de l'altération la plus récente	
MQCA_CHANNEL_AUTO_DEF_EXIT	MQ_EXIT_NAME_LENGTH	Nom d'exit de définition de canal automatique	
MQCA_CHINIT_SERVICE_PARM		Réservé à l'utilisation par IBM	
MQCA_CLUSTER_WORKLOAD_DATA	MQ_EXIT_DATA_LENGTH	Données transmises à l'exit de charge de travail du cluster	

Tableau 552. Sélecteurs d'attribut MQINQ pour le gestionnaire de files d'attente (suite)


Sélecteur	Longueur de la zone	Description	Remarque
MQCA_CLUSTER_WORKLOAD_EXIT	MQ_EXIT_NAME_LENGTH	Nom de l'exit de charge de travail de cluster	
MQCA_COMMAND_INPUT_Q_NAME	MQ_Q_NAME_LENGTH	Nom de la file d'attente d'entrée des commandes système	
MQCA_CUSTOM	MQ_CUSTOM_LENGTH	Attribut personnalisé pour les nouvelles fonctions	
MQCA_DEAD_LETTER_Q_NAME	MQ_Q_NAME_LENGTH	Nom de la file d'attente de rebut	
MQCA_DEF_XMIT_Q_NAME	MQ_Q_NAME_LENGTH	Nom de file d'attente de transmission par défaut	
MQCA_DNS_GROUP	MQ_DNS_GROUP_NAME_LENGTH	Nom du groupe du programme d'écoute TCP qui gère les transmissions entrantes pour le groupe de partage de files d'attente à joindre. Le nom s'applique lors de l'utilisation des services de nom de domaine dynamique de Workload Manager.	z/OS
MQCA_IGQ_USER_ID	MQ_USER_ID_LENGTH	Identificateur utilisateur de la mise en file d'attente intra-groupe	z/OS
 MQCA_INITIAL_KEY	MQ_INITIAL_KEY_LENGTH	Clé initiale du système de protection par mot de passe	Renvoie *** *** ** s'il n'est pas vide, ou vide si la clé initiale par défaut est utilisée
MQCA_INSTALLATION_DESC	MQ_INSTALLATION_DESC_LENGTH	Description de l'installation associée	Non z/OS NON IBM i

Tableau 552. Sélecteurs d'attribut MQINQ pour le gestionnaire de files d'attente (suite)

Sélecteur	Longueur de la zone	Description	Remarque
MQCA_INSTALLATION_NAME	MQ_INSTALLATION_NAME_LENGTH	Nom de l'installation associée au gestionnaire de files d'attente	Non z/OS · NON IBM i
MQCA_INSTALLATION_PATH	MQ_INSTALLATION_PATH_LENGTH	Chemin dans lequel le IBM MQ associé est installé	Non z/OS · NON IBM i
MQCA_LU_GROUP_NAME	MQ_LU_NAME_LENGTH	Nom d'unité logique générique du programme d'écoute d'unité logique 6.2 qui gère les transmissions entrantes pour le groupe de partage de files d'attente à utiliser	z/OS
MQCA_LU_NAME	MQ_LU_NAME_LENGTH	Nom de l'unité logique à utiliser pour les transmissions LU 6.2 sortantes. Définissez ce nom sur la même unité logique que celle utilisée par le programme d'écoute pour les transmissions entrantes	z/OS
MQCA_LU62_ARM_SUFFIX	MQ_ARM_SUFFIX_LENGTH	Suffixe du SYS1 . PARMLIB membre APPCPM <i>xx</i> , qui désigne la LUADD pour cet initiateur de canal	z/OS
MQCA_PARENT	MQ_Q_MGR_NAME_LENGTH	Nom d'un gestionnaire de files d'attente connecté de manière hiérarchique qui est désigné comme parent de ce gestionnaire de files d'attente	
MQCA_Q_MGR_DESC	MQ_Q_MGR_DESC_LENGTH	Description du gestionnaire de files d'attente	
MQCA_Q_MGR_IDENTIFIER	MQ_Q_MGR_IDENTIFIER_LENGTH	Identificateur du gestionnaire de files d'attente (H)	
MQCA_Q_MGR_NAME	MQ_Q_MGR_NAME_LENGTH	Nom du gestionnaire de files d'attente local	
MQCA_QSG_NAME	MQ_QSG_NAME_LENGTH	Nom du groupe de partage de files d'attente	z/OS
MQCA_REPOSITORY_NAME	MQ_CLUSTER_NAME_LENGTH	Nom du cluster pour lequel le gestionnaire de files d'attente fournit des services de référentiel	
MQCA_REPOSITORY_NAMELIST	MQ_NAMELIST_NAME_LENGTH	Nom de l'objet liste de noms contenant les noms des clusters pour lesquels le gestionnaire de files d'attente fournit des services de référentiel	



Tableau 552. Sélecteurs d'attribut MQINQ pour le gestionnaire de files d'attente (suite)


Sélecteur	Longueur de la zone	Description	Remarque
 MQCA_SSL_KEY_REPO_PASSWORD	MQ_SSL_ENCRYPT_KEY_REPO_PWD_LEN	Mot de passe du référentiel de clés	Renvoie *** *** ** s'il n'est pas vide, ou vide s'il n'est pas défini Chiffré lorsqu'il est défini avant le stockage.
MQCA_TCP_NAME	MQ_TCP_NAME_LENGTH	Nom du système TCP/IP que vous utilisez	z/OS
MQIA_ACCOUNTING_CONNOVERRIDE	MQLONG	Remplacer les paramètres de comptabilité	Non z/OS
MQIA_ACCOUNTING_INTERVAL	MQLONG	Fréquence d'écriture des enregistrements comptables intermédiaires	Non z/OS
MQIA_ACCOUNTING_MQI	MQLONG	Contrôle la collecte des informations de comptabilité pour les données MQI	Non z/OS
MQIA_ACCOUNTING_Q	MQLONG	Contrôle la collecte des informations de comptabilité pour les files d'attente	Non z/OS
MQIA_ACTIVE_CHANNELS	MQLONG	Nombre maximal de canaux pouvant être actifs à tout moment	z/OS
MQIA_ADOPTNEWMCA_CHECK	MQLONG	Éléments vérifiés pour déterminer s'il convient d'adopter un agent MCA. La vérification est effectuée lorsqu'un nouveau canal entrant portant le même nom qu'un agent MCA déjà actif est détecté.	z/OS

Tableau 552. Sélecteurs d'attribut MQINQ pour le gestionnaire de files d'attente (suite)

Sélecteur	Longueur de la zone	Description	Remarque
MQIA_ADOPTNEWMCA_INTERVAL	MQLONG	Durée, en secondes, pendant laquelle le nouveau canal attend l'arrêt du canal orphelin	Non z/OS
MQIA_ADOPTNEWMCA_TYPE	MQLONG	Indique s'il faut redémarrer automatiquement une instance orpheline d'un agent MCA d'un type de canal particulier lorsqu'une nouvelle demande de canal entrant correspondant aux paramètres AdoptNewMCACheck est détectée	z/OS
MQIA_AUTHORITY_EVENT	MQLONG	Attribut de contrôle des événements de droits d'accès	Non z/OS
MQIA_BRIDGE_EVENT	MQLONG	Attribut de contrôle pour les événements de pont IMS	z/OS
MQIA_CHANNEL_AUTO_DEF	MQLONG	Attribut de contrôle pour la définition de canal automatique	Non z/OS
MQIA_CHANNEL_AUTO_DEF_EVENT	MQLONG	Attribut de contrôle pour les événements de définition de canal automatique	Non z/OS
MQIA_CHANNEL_EVENT	MQLONG	Attribut de contrôle pour les événements de canal	
MQIA_CHINIT_ADAPTERS	MQLONG	Nombre de sous-tâches d'adaptateur à utiliser pour le traitement des appels IBM MQ	z/OS
MQIA_CHINIT_DISPATCHERS	MQLONG	Nombre de répartiteurs à utiliser pour l'initiateur de canal	z/OS
MQIA_CHINIT_TRACE_AUTO_START	MQLONG	Indique si la trace de l'initialisateur de canal doit être démarrée automatiquement	z/OS
MQIA_CHINIT_TRACE_TABLE_SIZE	MQLONG	Taille de l'espace des données de trace (en Mo) de l'initiateur de canal	z/OS
MQIA_CLUSTER_WORKLOAD_LENGTH	MQLONG	Longueur de la charge de travail du cluster.	
MQIA_CLWL_MRU_CHANNELS	MQLONG	Nombre de canaux les plus récemment utilisés pour l'équilibrage de charge de cluster	
MQIA_CLWL_USEQ	MQLONG	Utilisation des files d'attente éloignées	
MQIA_CODED_CHAR_SET_ID	MQLONG	Identificateur de jeu de caractères codés	
MQIA_COMMAND_EVENT	MQLONG	Attribut de contrôle pour les événements de commande	

Tableau 552. Sélecteurs d'attribut MQINQ pour le gestionnaire de files d'attente (suite)

Sélecteur	Longueur de la zone	Description	Remarque
MQIA_COMMAND_LEVEL	MQLONG	Niveau de commande pris en charge par le gestionnaire de files d'attente	
MQIA_CONFIGURATION_EVENT	MQLONG	Attribut de contrôle pour les événements de configuration	Non z/OS
MQIA_DEF_CLUSTER_XMIT_Q_TYPE	MQLONG	Type de file d'attente de transmission par défaut à utiliser pour les canaux émetteurs de cluster.	
MQIA_DIST_LISTS	MQLONG	Prise en charge de la liste de diffusion	Non z/OS
MQIA_DNS_WLM	MQLONG	Indique si le programme d'écoute TCP qui gère les transmissions entrantes pour le groupe de partage de files d'attente s'enregistre auprès de Workload Manager for Dynamic Domain Name Services	z/OS
MQIA_EXPIRY_INTERVAL	MQLONG	Intervalle entre les analyses des messages arrivés à expiration	z/OS
MQIA_GROUP_UR	MQLONG	Attribut de contrôle indiquant si les unités de récupération GROUP sont activées pour ce gestionnaire de files d'attente. La disposition de l'unité de récupération GROUP n'est disponible que si le gestionnaire de files d'attente est membre d'un groupe de partage de files d'attente	z/OS
MQIA_IGQ_PUT_AUTHORITY	MQLONG	Droit d'insertion de la mise en file d'attente intragroupe	z/OS
MQIA_INHIBIT_EVENT	MQLONG	Attribut de contrôle pour les événements d'interdiction	Non z/OS
MQIA_INTRA_GROUP_queuing	MQLONG	Prise en charge de la mise en file d'attente intra-groupe	z/OS
MQIA_LISTENER_TIMER	MQLONG	Intervalle (en secondes) entre les tentatives de redémarrage du programme d'écoute par IBM MQ en cas d'échec d'APPC ou de TCP/IP.	z/OS
MQIA_LOCAL_EVENT	MQLONG	Attribut de contrôle pour les événements locaux	Non z/OS
MQIA_LOGGER_EVENT	MQLONG	Attribut de contrôle pour les événements d'interdiction	Non z/OS
MQIA_LU62_CHANNELS	MQLONG	Nombre maximal de canaux pouvant être en cours ou de clients pouvant être connectés à l'aide du protocole de transmission LU 6.2	z/OS

Tableau 552. Sélecteurs d'attribut MQINQ pour le gestionnaire de files d'attente (suite)


Sélecteur	Longueur de la zone	Description	Remarque
MQIA_MSG_MARK_BROWSE_INTERVAL	MQLONG	Intervalle de temps (en millisecondes) après lequel le gestionnaire de files d'attente peut supprimer automatiquement une marque des messages de navigation.   <b>Avertissement :</b> Vous ne devez pas définir cette valeur en dessous de la valeur par défaut de 5000.	
MQIA_MAX_CHANNELS	MQLONG	Nombre maximal de canaux pouvant être en cours (y compris les canaux de connexion serveur avec des clients connectés)	z/OS
MQIA_MAX_HANDLES	MQLONG	Nombre maximal de descripteurs	
MQIA_MAX_MSG_LENGTH	MQLONG	Longueur maximale des messages	
MQIA_MAX_PRIORITY	MQLONG	Priorité maximum	
MQIA_MAX_UNCOMMITTED_MESSAGES	MQLONG	Nombre maximal de messages non validés dans une unité de travail	
MQIA_OUTBOUND_PORT_MAX	MQLONG	Avec MQIA_OUTBOUND_PORT_MIN, définit la plage de numéros de port à utiliser lors de la liaison des canaux sortants	z/OS
MQIA_OUTBOUND_PORT_MIN	MQLONG	Avec MQIA_OUTBOUND_PORT_MAX, définit la plage de numéros de port à utiliser lors de la liaison des canaux sortants	z/OS
MQIA_PERFORMANCE_EVENT	MQLONG	Attribut de contrôle pour les événements de performances	Non z/OS
MQIA_PLATFORM	MQLONG	Plateforme sur laquelle réside le gestionnaire de files d'attente	
MQIA_PROT_POLICY_CAPABILITY	MQLONG	Indique si les fonctions de sécurité de Advanced Message Security sont disponibles pour un gestionnaire de files d'attente.	
MQIA_PUBSUB_MAXMSG_RETRY_COUNT	MQLONG	Nombre de tentatives de traitement d'un message de commande ayant échoué sous le point de synchronisation	

Tableau 552. Sélecteurs d'attribut MQINQ pour le gestionnaire de files d'attente (suite)

Sélecteur	Longueur de la zone	Description	Remarque
MQIA_PUBSUB_MODE	MQLONG	Indique si le moteur de publication / abonnement et l'interface de publication / abonnement en file d'attente sont en cours d'exécution. Les applications de publication ou d'abonnement à l'aide de l'interface de programmation d'application requièrent le moteur de publication / abonnement. Les files d'attente surveillées par l'interface de publication / abonnement en file d'attente requièrent l'exécution de l'interface de publication / abonnement en file d'attente.	
MQIA_PUBSUB_NP_MSG	MQLONG	Indique s'il faut supprimer (ou conserver) un message d'entrée non distribué	
MQIA_PUBSUB_NP_RESP	MQLONG	Contrôle le comportement des messages de réponse non livrés.	
MQIA_PUBSUB_SYNC_PT	MQLONG	Indique si seuls les messages persistants (ou tous les messages) sont traités sous le point de synchronisation	
MQIA_QMGR_CFCONLOS	MQLONG	Indique l'action à effectuer lorsque le gestionnaire de files d'attente perd la connectivité à la structure d'administration ou à toute structure d'unité de couplage avec CFCONLOS défini sur ASQMGR	z/OS
MQIA_RECEIVE_TIMEOUT	MQLONG	Durée approximative pendant laquelle un canal TCP/IP attend de recevoir des données, y compris des signaux de présence, de la part de son partenaire, avant de revenir à l'état inactif. La valeur est numérique, qualifiée par MQIA_RECEIVE_TIMEOUT_TYPE.	z/OS
MQIA_RECEIVE_TIMEOUT_MIN	MQLONG	Durée minimale pendant laquelle un canal TCP/IP attend de recevoir des données, y compris des signaux de présence, de la part de son partenaire, avant de revenir à l'état inactif	z/OS

Tableau 552. Sélecteurs d'attribut MQINQ pour le gestionnaire de files d'attente (suite)

Sélecteur	Longueur de la zone	Description	Remarque
MQIA_RECEIVE_TIMEOUT_TYPE	MQLONG	Durée approximative pendant laquelle un canal TCP/IP attend de recevoir des données, y compris des signaux de présence, de la part de son partenaire, avant de revenir à l'état inactif. MQIA_RECEIVE_TIMEOUT_TYPE est le qualificateur appliqué à MQIA_RECEIVE_TIMEOUT.	z/OS
MQIA_REMOTE_EVENT	MQLONG	Attribut de contrôle pour les événements distants	Non z/OS
MQIA_SECURITY_CASE	MQLONG	Cas des profils de sécurité	z/OS
MQIA_SSL_EVENT	MQLONG	Attribut de contrôle pour les événements de canal	
MQIA_SSL_FIPS_REQUIRED	MQLONG	Utiliser uniquement des algorithmes certifiés FIPS pour la cryptographie	
MQIA_SSL_RESET_COUNT	MQLONG	Nombre de réinitialisations de clé TLS	
MQIA_START_STOP_EVENT	MQLONG	Attribut de contrôle pour les événements d'arrêt de démarrage	Non z/OS
MQIA_STATISTICS_AUTO_CLUSTER	MQLONG	Contrôle la collecte des informations de surveillance des statistiques pour les canaux émetteurs de cluster	
MQIA_STATISTICS_CHANNEL	MQLONG	Contrôle la collecte des données statistiques pour les canaux	
MQIA_STATISTICS_INTERVAL	MQLONG	Fréquence d'écriture des données de surveillance des statistiques	Non z/OS
MQIA_STATISTICS_MQI	MQLONG	Contrôle la collecte des informations de surveillance des statistiques pour le gestionnaire de files d'attente	Non z/OS
MQIA_STATISTICS_Q	MQLONG	Contrôle la collecte des données statistiques pour les files d'attente	Non z/OS
MQIA_SYNCPOINT	MQLONG	disponibilité des points de synchronisation	
MQIA_TCP_CHANNELS	MQLONG	Nombre maximal de canaux pouvant être en cours ou de clients pouvant être connectés à l'aide du protocole de transmission TCP/IP	z/OS
MQIA_TCP_KEEP_ALIVE	MQLONG	Indique s'il faut utiliser la fonction TCP KEEPALIVE pour vérifier que l'autre extrémité de la connexion est toujours disponible	z/OS

Tableau 552. Sélecteurs d'attribut MQINQ pour le gestionnaire de files d'attente (suite)

Sélecteur	Longueur de la zone	Description	Remarque
MQIA_TCP_STACK_TYPE	MQLONG	Indique si l'initiateur de canal peut utiliser uniquement l'espace adresse TCP/IP spécifié dans TCPNAME ou s'il peut éventuellement se connecter à une adresse TCP/IP sélectionnée	z/OS
MQIA_TRACE_ROUTE_RECORDING	MQLONG	Contrôle l'enregistrement des informations de trace-route	z/OS
MQIA_TREE_LIFE_TIME	MQLONG	Durée de vie des rubriques non administratives inutilisées	
MQIA_TRIGGER_INTERVAL	MQLONG	Intervalle de déclenchement	

### IntAttrCount

Type: MQLONG -entrée

Il s'agit du nombre d'éléments du tableau *IntAttrs*. Zéro est une valeur valide.

Si *IntAttrCount* est au moins le nombre de sélecteurs MQIA\_\* dans le paramètre **Selectors**, tous les attributs entiers demandés sont renvoyés.

### IntAttrs

Type: MQLONG x *IntAttrCount* -sortie

Il s'agit d'un tableau de valeurs d'attribut d'entier *IntAttrCount*.

Les valeurs d'attribut entières sont renvoyées dans le même ordre que les sélecteurs MQIA\_\* dans le paramètre **Selectors**. Si le tableau contient plus d'éléments que le nombre de sélecteurs MQIA\_\*, les éléments excédentaires restent inchangés.

Si *Hobj* représente une file d'attente, mais qu'un sélecteur d'attribut ne s'applique pas à ce type de file d'attente, la valeur spécifique MQIAV\_NOT\_APPLICABLE est renvoyée. Il est renvoyé pour l'élément correspondant dans le tableau *IntAttrs*.

Si le paramètre **IntAttrCount** ou **SelectorCount** a pour valeur zéro, il n'est pas fait référence à *IntAttrs*. Dans ce cas, l'adresse de paramètre transmise par les programmes écrits en assembleur C ou S/390 peut être null.

### CharAttrLongueur

Type: MQLONG -entrée

Il s'agit de la longueur en octets du paramètre **CharAttrs**.

*CharAttrLongueur* doit être au moins la somme des longueurs des attributs de caractères demandés (voir Sélecteurs). Zéro est une valeur valide.

### CharAttrs

Type: MQCHAR x *CharAttrLength* -sortie

Il s'agit de la mémoire tampon dans laquelle les attributs de caractère sont renvoyés, concaténés. La longueur de la mémoire tampon est indiquée par le paramètre **CharAttrLength**.

Les attributs de caractère sont renvoyés dans le même ordre que les sélecteurs MQCA\_\* dans le paramètre **Selectors**. La longueur de chaque chaîne d'attribut est fixe pour chaque attribut (voir Sélecteurs) et la valeur qu'elle contient est complétée à droite par des blancs si nécessaire. Vous pouvez fournir une mémoire tampon plus grande que nécessaire pour contenir tous les attributs de caractères demandés et le remplissage. Les octets au-delà de la dernière valeur d'attribut renvoyée restent inchangés.

Si *Hobj* représente une file d'attente, mais qu'un sélecteur d'attribut ne s'applique pas à ce type de file d'attente, une chaîne de caractères composée entièrement d'astérisques (\*) est renvoyée. L'astérisque est renvoyé en tant que valeur de cet attribut dans *CharAttr*.

Si le paramètre *CharAttrLength* ou **SelectorCount** a pour valeur zéro, il n'est pas fait référence à *CharAttr*. Dans ce cas, l'adresse de paramètre transmise par les programmes écrits en assembleur C ou S/390 peut être null.

### CompCode

Type: MQLONG -sortie

Code achèvement:

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_WARNING**

Avertissement (achèvement partiel).

#### **MQCC\_FAILED**

Echec de l'appel.

### Motif

Type : MQLONG - sortie

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

( 0, X'000 ') Aucune raison de signaler.

Si *CompCode* est MQCC\_WARNING:

#### **MQRC\_CHAR\_ATTRS\_TOO\_SHORT**

( 2008, X'7D8 ') Espace insuffisant pour les attributs de caractères.

#### **MQRC\_INT\_ATTR\_COUNT\_TOO\_SMALL**

( 2022, X'7E6 ') Espace insuffisant pour les attributs de type entier.

#### **MQRC\_SELECTOR\_NOT\_FOR\_TYPE**

( 2068, X'814 ') Sélecteur non applicable au type de file d'attente.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_ADAPTER\_NOT\_AVAILABLE**

( 2204, X'89C ') Adaptateur non disponible.

#### **MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

( 2130, X'852 ') Impossible de charger le module de service de l'adaptateur.

#### **MQRC\_API\_EXIT\_ERROR**

( 2374, X'946 ') L'exit d'API a échoué.

#### **MQRC\_API\_EXIT\_LOAD\_ERROR**

( 2183, X'887 ') Impossible de charger l'exit d'API.

#### **MQRC\_ASID\_MISMATCH**

( 2157, X'86D ') Les ID espace adresse principal et principal sont différents.

#### **MQRC\_CALL\_IN\_PROGRESS**

( 2219, X'8AB ') Appel MQI entré avant la fin de l'appel précédent.

#### **MQRC\_CF\_STRUC\_FAILED**

( 2373, X'945 ') Echec de la structure de l'unité de couplage.

#### **MQRC\_CF\_STRUC\_IN\_USE**

( 2346, X'92A ') Structure d'unité de couplage en cours d'utilisation.

#### **MQRC\_CHAR\_ATTR\_LENGTH\_ERROR**

( 2006, X'7D6 ') La longueur des attributs de caractère n'est pas valide.

#### **MQRC\_CHAR\_ATTRS\_ERROR**

( 2007, X'7D7 ') Chaîne d'attributs de caractères non valide.



**MQRC\_CICS\_WAIT\_FAILED**  
(2140, X'85C') Demande d'attente rejetée par CICS.

**MQRC\_CONNECTION\_BROKEN**  
(2009, X'7D9') La connexion au gestionnaire de files d'attente a été perdue.

**MQRC\_CONNECTION\_NOT\_AUTHORIZED**  
(2217, X'8A9') Non autorisé pour la connexion.

**MQRC\_CONNECTION\_STOPPING**  
(2203, X'89B') Arrêt de la connexion.

**MQRC\_HCONN\_ERROR**  
(2018, X'7E2') Descripteur de connexion incorrect.

**MQRC\_HOBJ\_ERROR**  
(2019, X'7E3') Descripteur d'objet non valide.

**MQRC\_INT\_ATTR\_COUNT\_ERROR**  
(2021, X'7E5') Nombre d'attributs de type entier non valide.

**MQRC\_INT\_ATTRS\_ARRAY\_ERROR**  
(2023, X'7E7') Tableau d'attributs d'entier non valide.

**MQRC\_NOT\_OPEN\_FOR\_INQUIRE**  
(2038, X'7F6') File d'attente non ouverte pour interrogation.

**MQRC\_OBJECT\_CHANGED**  
(2041, X'7F9') La définition d'objet a été modifiée depuis son ouverture.

**MQRC\_OBJECT\_DAMAGED**  
(2101, X'835') Objet endommagé.

**MQRC\_PAGESET\_ERROR**  
(2193, X'891') Erreur lors de l'accès à l'ensemble de données de l'ensemble de pages.

**MQRC\_Q\_DELETED**  
(2052, X'804') File d'attente supprimée.

**MQRC\_Q\_MGR\_NAME\_ERROR**  
(2058, X'80A') Le nom du gestionnaire de files d'attente est incorrect ou inconnu.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**  
(2059, X'80B') Le gestionnaire de files d'attente n'est pas disponible pour la connexion.

**MQRC\_Q\_MGR\_STOPPING**  
(2162, X'872') Arrêt du gestionnaire de files d'attente.

**MQRC\_RESOURCE\_PROBLEM**  
(2102, X'836') Ressources système disponibles insuffisantes.

**MQRC\_SELECTOR\_COUNT\_ERROR**  
(2065, X'811') Nombre de sélecteurs non valide.

**MQRC\_SELECTOR\_ERROR**  
(2067, X'813') Sélecteur d'attribut non valide.

**MQRC\_SELECTOR\_LIMIT\_EXCEEDED**  
(2066, X'812') Nombre de sélecteurs trop élevé.

**MQRC\_STORAGE\_NOT\_AVAILABLE**  
(2071, X'817') Mémoire disponible insuffisante.

**MQRC\_SUPPRESSED\_BY\_EXIT**  
(2109, X'83D') Appel supprimé par le programme d'exit.

**MQRC\_UNEXPECTED\_ERROR**  
(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#)

## Remarques sur l'utilisation

1. Les valeurs renvoyées sont un instantané des attributs sélectionnés. Il n'est pas garanti que les attributs restent les mêmes avant que l'application puisse agir sur les valeurs renvoyées.
2. Lorsque vous ouvrez une file d'attente modèle, une file d'attente locale dynamique est créée. Une file d'attente locale dynamique est créée même si vous ouvrez la file d'attente modèle pour en savoir plus sur ses attributs.

Les attributs de la file d'attente dynamique sont en grande partie les mêmes que ceux de la file d'attente modèle lors de la création de la file d'attente dynamique. Si vous utilisez ensuite l'appel MQINQ sur cette file d'attente, le gestionnaire de files d'attente renvoie les attributs de la file d'attente dynamique et non ceux de la file d'attente modèle. Pour plus de détails sur les attributs de la file d'attente modèle hérités par la file d'attente dynamique, voir [Tableau 561](#), à la [page 878](#).

3. Si l'objet demandé est une file d'attente alias, les valeurs d'attribut renvoyées par l'appel MQINQ sont les attributs de la file d'attente alias. Il ne s'agit pas des attributs de la file d'attente de base ou de la rubrique dans laquelle l'alias est résolu.
4. Si l'objet demandé est une file d'attente de cluster, les attributs pouvant être demandé dépendent de la manière dont la file d'attente est ouverte:

- Vous pouvez ouvrir une file d'attente de cluster pour interroger plus une ou plusieurs opérations d'entrée, de navigation ou de définition. Pour ce faire, il doit exister une instance locale de la file d'attente de cluster pour que l'ouverture aboutisse. Dans ce cas, les attributs qui peuvent être renseignés sont les attributs qui sont valides pour les files d'attente locales.

Si la file d'attente de cluster est ouverte pour interrogation sans entrée, exploration ou définition spécifiée, l'appel renvoie le code achèvement MQCC\_WARNING et le code anomalie MQRC\_SELECTOR\_NOT\_FOR\_TYPE (2068) si vous tentez d'interroger des attributs qui sont valides uniquement pour les files d'attente locales et non pour les files d'attente de cluster.

- Vous pouvez ouvrir une file d'attente de cluster pour l'interrogation lors de la transmission du nom de gestionnaire de files d'attente de base du gestionnaire de files d'attente connecté.

Pour ce faire, il doit exister une instance locale de la file d'attente de cluster pour que l'ouverture aboutisse. Si le gestionnaire de files d'attente de base n'est pas transmis, l'appel renvoie le code achèvement MQCC\_WARNING et le code anomalie MQRC\_SELECTOR\_NOT\_FOR\_TYPE (2068) si vous tentez d'interroger des attributs qui sont valides uniquement pour les files d'attente locales et non pour les files d'attente de cluster.

- Si la file d'attente de cluster est ouverte pour l'interrogation seule ou pour l'interrogation et la sortie, seuls les attributs répertoriés peuvent être utilisés pour l'interrogation. L'attribut **QType** a la valeur MQQT\_CLUSTER dans ce cas:

- MQCA\_Q\_DESC
- MQCA\_Q\_NAME
- MQIA\_DEF\_BIND
- MQIA\_DEF\_PERSISTENCE
- MQIA\_DEF\_PRIORITY
- MQIA\_INHIBIT\_PUT
- MQIA\_Q\_TYPE

Vous pouvez ouvrir la file d'attente de cluster sans liaison fixe. Vous pouvez l'ouvrir avec MQOO\_BIND\_NOT\_FIXED spécifié dans l'appel MQOPEN. Vous pouvez également spécifier MQOO\_BIND\_AS\_Q\_DEF et définir l'attribut **DefBind** de la file d'attente sur MQBND\_BIND\_NOT\_FIXED. Si vous ouvrez une file d'attente de cluster sans liaison fixe, les appels MQINQ successifs de la file d'attente peuvent interroger des instances différentes de la file d'attente de cluster. Toutefois, il est typique que toutes les instances aient les mêmes valeurs d'attribut.

- Un objet file d'attente alias peut être défini pour un cluster. Etant donné que TARGTYPE et TARGET ne sont pas des attributs de cluster, le processus exécutant un processus MQOPEN sur la file d'attente alias ne connaît pas l'objet dans lequel l'alias est résolu.

Lors de la MQOPENinitiale, la file d'attente alias se résout en un gestionnaire de files d'attente et en une file d'attente dans le cluster. La résolution de nom se produit à nouveau au niveau du gestionnaire de files d'attente éloignées et c'est ici que le TARGTPYE de la file d'attente alias est résolu.

Si la file d'attente alias se résout en alias de rubrique, la publication des messages insérés dans la file d'attente alias a lieu sur ce gestionnaire de files d'attente éloigné.

Voir [Files d'attente de cluster](#)

5. Vous pouvez demander un certain nombre d'attributs, puis en définir certains à l'aide de l'appel MQSET . Pour programmer une interrogation et une définition efficaces, positionnez les attributs à définir au début des tableaux de sélecteurs. Dans ce cas, les mêmes tableaux avec des effectifs réduits peuvent être utilisés pour MQSET.
6. Si plusieurs situations d'avertissement se produisent (voir le paramètre **CompCode** ), le code anomalie renvoyé est le premier de la liste suivante qui s'applique:
  - a. MQRC\_SELECTOR\_NOT\_FOR\_TYPE
  - b. MQRC\_INT\_ATTR\_COUNT\_TOO\_SMALL
  - c. MQRC\_CHAR\_ATTRS\_TOO\_SHORT
7. La rubrique suivante contient des informations sur les attributs d'objet:
  - [«Attributs des files d'attente»](#), à la page 875
  - [«Attributs des listes de noms»](#), à la page 911
  - [«Attributs des définitions de processus»](#), à la page 913
  - [«Attributs du gestionnaire de files d'attente»](#), à la page 835

## Appel C

```
MQINQ (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs,  
CharAttrLength, CharAttrs, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;           /* Connection handle */  
MQHOBJ   Hobj;           /* Object handle */  
MQLONG   SelectorCount; /* Count of selectors */  
MQLONG   Selectors[n];  /* Array of attribute selectors */  
MQLONG   IntAttrCount;  /* Count of integer attributes */  
MQLONG   IntAttrs[n];   /* Array of integer attributes */  
MQLONG   CharAttrLength; /* Length of character attributes buffer */  
MQCHAR   CharAttrs[n]; /* Character attributes */  
MQLONG   CompCode;      /* Completion code */  
MQLONG   Reason;        /* Reason code qualifying CompCode */
```

## Appel COBOL

```
CALL 'MQINQ' USING HCONN, HOBJ, SELECTORCOUNT, SELECTORS-TABLE,  
INTATTRCOUNT, INTATTRS-TABLE, CHARATTRLENGTH,  
CHARATTRS, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle  
01 HCONN          PIC S9(9) BINARY.  
** Object handle  
01 HOBJ           PIC S9(9) BINARY.  
** Count of selectors  
01 SELECTORCOUNT PIC S9(9) BINARY.  
** Array of attribute selectors  
01 SELECTORS-TABLE.
```

```

02 SELECTORS          PIC S9(9) BINARY OCCURS n TIMES.
** Count of integer attributes
01 INTATTRCOUNT    PIC S9(9) BINARY.
** Array of integer attributes
01 INTATTRS-TABLE.
02 INTATTRS          PIC S9(9) BINARY OCCURS n TIMES.
** Length of character attributes buffer
01 CHARATTRLENGTH    PIC S9(9) BINARY.
** Character attributes
01 CHARATTRS          PIC X(n).
** Completion code
01 COMPCODE           PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON             PIC S9(9) BINARY.

```

## Appel PL/I

```

call MQINQ (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount,
            IntAttrs, CharAttrLength, CharAttrs, CompCode, Reason);

```

Déclarez les paramètres comme suit :

```

dcl Hconn          fixed bin(31); /* Connection handle */
dcl Hobj           fixed bin(31); /* Object handle */
dcl SelectorCount  fixed bin(31); /* Count of selectors */
dcl Selectors(n)   fixed bin(31); /* Array of attribute selectors */
dcl IntAttrCount   fixed bin(31); /* Count of integer attributes */
dcl IntAttrs(n)    fixed bin(31); /* Array of integer attributes */
dcl CharAttrLength fixed bin(31); /* Length of character attributes
buffer */
dcl CharAttrs      char(n);       /* Character attributes */
dcl CompCode       fixed bin(31); /* Completion code */
dcl Reason         fixed bin(31); /* Reason code qualifying
CompCode */

```

## Appel d'assembleur de haut niveau

```

CALL MQINQ, (HCONN,HOBJ,SELECTORCOUNT,SELECTORS,INTATTRCOUNT, X
            INTATTRS,CHARATTRLENGTH,CHARATTRS,COMPCODE,REASON)

```

Déclarez les paramètres comme suit :

```

HCONN          DS F      Connection handle
HOBJ           DS F      Object handle
SELECTORCOUNT DS F      Count of selectors
SELECTORS      DS (n)F   Array of attribute selectors
INTATTRCOUNT  DS F      Count of integer attributes
INTATTRS       DS (n)F   Array of integer attributes
CHARATTRLENGTH DS F      Length of character attributes buffer
CHARATTRS      DS CL(n)  Character attributes
COMPCODE       DS F      Completion code
REASON         DS F      Reason code qualifying COMPCODE

```

## Appel Visual Basic

```

MQINQ Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs,
      CharAttrLength, CharAttrs, CompCode, Reason

```

Déclarez les paramètres comme suit :

```

Dim Hconn          As Long 'Connection handle'
Dim Hobj           As Long 'Object handle'
Dim SelectorCount  As Long 'Count of selectors'
Dim Selectors      As Long 'Array of attribute selectors'

```

Dim IntAttrCount	As Long	'Count of integer attributes'
Dim IntAttrs	As Long	'Array of integer attributes'
Dim CharAttrLength	As Long	'Length of character attributes buffer'
Dim CharAttrs	As String	'Character attributes'
Dim CompCode	As Long	'Completion code'
Dim Reason	As Long	'Reason code qualifying CompCode'

## MQINQMP-Propriété de message d'interrogation

L'appel MQINQMP renvoie la valeur d'une propriété d'un message.

### Syntaxe

MQINQMP (*Hconn*, *Hmsg*, *InqPropOpts*, *Name*, *PropDesc*, *Type*, *ValueLength*, *Value*, *DataLength*, *CompCode*, *Reason*)

### Paramètres

#### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* doit correspondre au descripteur de connexion qui a été utilisé pour créer le descripteur de message spécifié dans le paramètre **Hmsg** .

Si le descripteur de message a été créé à l'aide de MQHC\_UNASSOCIATED\_HCONN, une connexion valide doit être établie sur l'unité d'exécution demandant une propriété du descripteur de message, sinon l'appel échoue avec MQRC\_CONNECTION\_BROKEN.

#### Msg

Type: MQHMSG-entrée

Il s'agit du descripteur de message à contrôler. La valeur a été renvoyée par un appel **MQCRTMH** précédent.

#### Options InqProp

Type: MQIMPO-entrée / sortie

Pour plus de détails, voir le type de données [MQIMPO](#) .

#### Nom

Type: MQCHARV-entrée / sortie

Nom de la propriété à interroger.

Si aucune propriété portant ce nom n'est trouvée, l'appel échoue avec le motif MQRC\_PROPERTY\_NOT\_AVAILABLE.

Vous pouvez utiliser le caractère générique pourcentage (%) à la fin du nom de la propriété. Le caractère générique correspond à zéro ou plusieurs caractères, y compris le point (.). Cela permet à une application d'interroger la valeur de nombreuses propriétés. Appelez MQINQMP avec l'option MQIMPO\_INQ\_FIRST pour obtenir la première propriété correspondante et à nouveau avec l'option MQIMPO\_INQ\_NEXT pour obtenir la propriété correspondante suivante. Lorsqu'aucune propriété correspondante n'est disponible, l'appel échoue avec MQRC\_PROPERTY\_NOT\_AVAILABLE. Si la zone *ReturnedName* de la structure *InqPropOpts* est initialisée avec une adresse ou un décalage pour le nom renvoyé de la propriété, cette opération est effectuée lors du retour de MQINQMP avec le nom de la propriété qui a été mise en correspondance. Si la zone *VSBufSize* de *ReturnedName* dans la structure *InqPropOpts* est inférieure à la longueur du nom de propriété renvoyé, le code achèvement est défini sur MQCC\_FAILED avec le motif MQRC\_PROPERTY\_NAME\_TOO\_BIG.

Les propriétés qui possèdent des synonymes connus sont renvoyées comme suit:

1. Propriétés avec le préfixe "mqps." sont renvoyés en tant que nom de propriété IBM MQ . Par exemple, "MQTopicString" est le nom renvoyé et non "mqps.Top"

2. Propriétés avec le préfixe "jms." ou "mcd." sont renvoyés en tant que nom de zone d'en-tête JMS , par exemple, "JMSExpiration" est le nom renvoyé et non "jms.Exp".
3. Propriétés avec le préfixe "usr." sont renvoyés sans ce préfixe, par exemple, "Color" est renvoyé à la place de "usr.Color".

Les propriétés avec des synonymes ne sont renvoyées qu'une seule fois.

Dans le langage de programmation C, les variables de macro suivantes sont définies pour l'interrogation de toutes les propriétés, puis de toutes les propriétés qui commencent par "usr.":

### **MQPROP\_INQUIRE\_ALL**

Consultez toutes les propriétés du message.

MQPROP\_INQUIRE\_ALL peut être utilisé de la manière suivante:

```
MQCHARV Name = {MQPROP_INQUIRE_ALL};
```

### **MQPROP\_INQUIRE\_ALL\_USR**

Consultez toutes les propriétés du message qui démarrent "usr.". Le nom renvoyé est renvoyé sans la valeur "usr." .

Si MQIMP\_INQ\_NEXT est spécifié mais que le nom a changé depuis l'appel précédent ou qu'il s'agit du premier appel, MQIMPO\_INQ\_FIRST est implicite.

Pour plus d'informations sur l'utilisation des noms de propriété, voir [Noms de propriété](#) et [Restrictions relatives aux noms de propriété](#) .

### **PropDesc**

Type: MQPD-sortie

Cette structure est utilisée pour définir les attributs d'une propriété, y compris ce qui se produit si la propriété n'est pas prise en charge, le contexte de message auquel la propriété appartient et les messages dans lesquels la propriété doit être copiée. Pour plus d'informations sur cette structure, voir [MQPD](#) .

### **Tapez**

Type: MQLONG-entrée / sortie

En cas de retour de l'appel MQINQMP, ce paramètre est défini sur le type de données *Valeur*. Le type de données peut être l'un des suivants:

#### **MQTYPE\_BOOLÉEN**

Une valeur booléenne.

#### **MQTYPE\_BYTE\_STRING**

une chaîne d'octets.

#### **MQTYPE\_INT8**

Entier signé de 8 bits.

#### **MQTYPE\_INT16**

Entier signé 16 bits.

#### **MQTYPE\_INT32**

Entier signé 32 bits.

#### **MQTYPE\_INT64**

Entier signé 64 bits.

#### **MQTYPE\_FLOAT32**

Nombre à virgule flottante 32 bits.

#### **MQTYPE\_FLOAT64**

Nombre en virgule flottante 64 bits.

#### **MQTYPE\_CHAINE**

Chaîne de caractères.

## **MQTYPE\_NULL**

La propriété existe mais a une valeur nulle.

Si le type de données de la valeur de propriété n'est pas reconnu, MQTYPE\_STRING est renvoyé et une représentation de chaîne de la valeur est placée dans la zone *Valeur*. Une représentation sous forme de chaîne du type de données est disponible dans la zone *TypeString* du paramètre *InqPropOpts*. Un code achèvement d'avertissement est renvoyé avec la raison MQRC\_PROP\_TYPE\_NOT\_SUPPORTED.

De plus, si l'option MQIMPO\_CONVERT\_TYPE est spécifiée, la conversion de la valeur de propriété est demandée. Utilisez *Type* comme entrée pour spécifier le type de données sous lequel vous souhaitez que la propriété soit renvoyée. Pour plus de détails sur la conversion de type de données, voir la description de l'option MQIMPO\_CONVERT\_TYPE de la structure MQIMPO.

Si vous ne demandez pas de conversion de type, vous pouvez utiliser la valeur suivante en entrée:

## **MQTYPE\_AS\_SET**

La valeur de la propriété est renvoyée sans convertir son type de données.

## **ValueLength**

Type : MQLONG - entrée

Longueur en octets de la zone *Valeur*. Spécifiez zéro pour les propriétés pour lesquelles vous n'avez pas besoin de la valeur renvoyée. Il peut s'agir de propriétés conçues par une application pour avoir une valeur nulle ou une chaîne vide. Spécifiez également zéro si l'option MQIMPO\_QUERY\_LENGTH a été spécifiée ; dans ce cas, aucune valeur n'est renvoyée.

## **Valeur**

Type: MQBYTEx *ValueLength* -sortie

Il s'agit de la zone devant contenir la valeur de la propriété interrogée. La mémoire tampon doit être alignée sur une limite appropriée pour la valeur renvoyée. Si vous ne le faites pas, une erreur peut se produire lors de l'accès ultérieur à la valeur.

Si *ValueLength* est inférieur à la longueur de la valeur de la propriété, la plus grande partie de la valeur de la propriété est déplacée dans *Valeur* et l'appel échoue avec le code achèvement MQCC\_FAILED et la raison MQRC\_PROPERTY\_VALEUR\_TOO\_BIG.

Le jeu de caractères des données dans *Valeur* est fourni par la zone ReturnedCCSID dans le paramètre Opts InqProp. Le codage des données dans *Valeur* est donné par la zone ReturnedEncoding dans le paramètre Opts InqProp.

Dans le langage de programmation C, le paramètre est déclaré comme un pointeur vers vide ; l'adresse de tout type de données peut être spécifiée comme paramètre.

Si le paramètre *ValueLength* est égal à zéro, *Value* n'est pas référencé et sa valeur transmise par les programmes écrits en assembleur C ou System/390 peut être null.

## **DataLength**

Type : MQLONG - sortie

Il s'agit de la longueur en octets de la valeur de propriété réelle renvoyée dans la zone *Valeur*.

Si *DataLength* est inférieur à la longueur de la valeur de la propriété, *DataLength* est toujours renseigné en cas de retour de l'appel MQINQMP. Cela permet à l'application de déterminer la taille de la mémoire tampon requise pour la valeur de propriété, puis d'émettre à nouveau l'appel avec une mémoire tampon de la taille appropriée.

Les valeurs suivantes peuvent également être renvoyées.

Si le paramètre *Type* est défini sur MQTYPE\_STRING ou MQTYPE\_BYTE\_STRING:

## **MQVL\_EMPTY\_CHAINE**

La propriété existe mais ne contient pas de caractères ni d'octets.

## **CompCode**

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

**MQCC\_OK**

Achèvement réussi.

**MQCC\_WARNING**

Avertissement (achèvement partiel).

**MQCC\_FAILED**

Echec de l'appel.

**Motif**

Type : MQLONG - sortie

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_WARNING:

**MQRC\_PROP\_NOM\_NON\_CONVERTI**

(2492, X'09BC') Nom de propriété renvoyé non converti.

**MQRC\_PROP\_VALEUR\_NOT\_CONVERTED**

(2466, X'09A2') Valeur de propriété non convertie.

**MQRC\_PROP\_TYPE\_NON\_PRIS en charge**

(2467, X'09A3') Le type de données de propriété n'est pas pris en charge.

**MQRC\_RFH\_FORMAT\_ERREUR**

(2421, X'0975') Un dossier MQRFH2 contenant des propriétés n'a pas pu être analysé.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'089C') Carte non disponible.

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'0852') Impossible de charger le module de service de l'adaptateur.

**MQRC\_ASID\_MISMATCH**

(2157, X'086D') Les ASID principal et principal diffèrent.

**MQRC\_BUFFER\_ERROR**

(2004, X'07D4') Paramètre de valeur non valide.

**MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'07D5') Paramètre de longueur de valeur non valide.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'07D9') La connexion au gestionnaire de files d'attente a été perdue.

**MQRC\_DATA\_LENGTH\_ERROR**

(2010, X'07DA') Paramètre de longueur de données non valide.

**MQRC\_IMPO\_ERREUR**

(2464, X'09A0') La structure des options de propriété de message d'interrogation n'est pas valide.

**ERREUR MQRC\_HMSG\_ERROR**

(2460, X'099C') Descripteur de message non valide.

**MQRC\_MSG\_HANDLE\_IN\_USE**

(2499, X'09C3') Descripteur de message déjà utilisé.

**MQRC\_OPTIONS\_ERROR**

(2046, X'07F8') Options non valides ou non cohérentes.

**MQRC\_PD\_ERREUR**

(2482, X'09B2') Structure de descripteur de propriété non valide.



**MQRC\_PROP\_CONV\_NOT\_SUPPORTED**

(2470, X'09A6') La conversion du type de données réel au type de données demandé n'est pas prise en charge.

**MQRC\_PROPERTY\_NAME\_ERROR**

(2442, X'098A') Nom de propriété non valide.

**MQRC\_PROPERTY\_NAME\_TOO\_BIG**

(2465, X'09A1') Nom de propriété trop grand pour la mémoire tampon de nom renvoyée.

**MQRC\_PROPERTY\_NOT\_AVAILABLE**

(2471, X'09A7) Propriété non disponible.

**MQRC\_PROPERTY\_VALEUR\_TOO\_BIG**

(2469, X'09A5') Valeur de propriété trop grande pour la zone Valeur.

**MQRC\_PROP\_NUMBER\_FORMAT\_ERROR**

(2472, X'09A8') Erreur de format numérique détectée dans les données de valeur.

**MQRC\_PROPERTY\_TYPE\_ERROR**

(2473, X'09A9') Type de propriété demandé non valide.

**MQRC\_SOURCE\_CCSID\_ERROR**

(2111, X'083F') Identificateur de jeu de caractères codés de nom de propriété non valide.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'0871') Mémoire disponible insuffisante.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'0893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Appel C

```
MQINQMP (Hconn, Hmsg, &InqPropOpts, &Name, &PropDesc, &Type,
ValueLength, Value, &DataLength, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN Hconn;          /* Connection handle */
MQHMSG  Hmsg;           /* Message handle */
MQIMPO  InqPropOpts;   /* Options that control the action of MQINQMP */
MQCHARV Name;          /* Property name */
MQPD    PropDesc;      /* Property descriptor */
MQLONG  Type;          /* Property data type */
MQLONG  ValueLength;   /* Length in bytes of the Value area */
MQBYTE  Value[n];      /* Area to contain the property value */
MQLONG  DataLength;    /* Length of the property value */
MQLONG  CompCode;      /* Completion code */
MQLONG  Reason;        /* Reason code qualifying CompCode */
```

## Appel COBOL

```
CALL 'MQINQMP' USING HCONN, HMSG, INQMSGOPTS, NAME, PROPDESC, TYPE,
VALUELENGTH, VALUE, DATALENGTH, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle
01 HCONN          PIC S9(9) BINARY.
** Message handle
01 HMSG           PIC S9(18) BINARY.
** Options that control the action of MQINQMP
01 INQMSGOPTS.
   COPY CMQIMPOV.
** Property name
```

```

01 NAME.
   COPY CMQCHRVV.
** Property descriptor
01 PROPDESC.
   COPY CMQPDV.
** Property data type
01 TYPE      PIC S9(9) BINARY.
** Length in bytes of the VALUE area
01 VALUELENGTH PIC S9(9) BINARY.
** Area to contain the property value
01 VALUE      PIC X(n).
** Length of the property value
01 DATALENGTH PIC S9(9) BINARY.
** Completion code
01 COMPCODE    PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON      PIC S9(9) BINARY.

```

## Appel PL/I

```

call MQINQMP (Hconn, Hmsg, InqPropOpts, Name, PropDesc, Type,
ValueLength, Value, DataLength, CompCode, Reason);

```

Déclarez les paramètres comme suit :

```

dcl Hconn      fixed bin(31); /* Connection handle */
dcl Hmsg       fixed bin(63); /* Message handle */
dcl InqPropOpts like MQIMPO; /* Options that control the action of MQINQMP */
dcl Name       like MQCHARV; /* Property name */
dcl PropDesc   like MQPD; /* Property descriptor */
dcl Type       fixed bin (31); /* Property data type */
dcl ValueLength fixed bin (31); /* Length in bytes of the Value area */
dcl Value      char (n); /* Area to contain the property value */
dcl DataLength fixed bin (31); /* Length of the property value */
dcl CompCode   fixed bin (31); /* Completion code */
dcl Reason     fixed bin (31); /* Reason code qualifying CompCode */

```

## Appel d'assembleur de haut niveau

```

CALL MQINQMP, (HCONN,HMSG,INQMSGOPTS,NAME,PROPDESC,TYPE,
VALUELENGTH,VALUE,DATALENGTH,COMPCODE,REASON)

```

Déclarez les paramètres comme suit :

HCONN	DS	F	Connection handle
HMSG	DS	D	Message handle
INQMSGOPTS	CMQIMPOA	,	Options that control the action of MQINQMP
NAME	CMQCHRVA	,	Property name
PROPDESC	CMQPDA	,	Property descriptor
TYPE	DS	F	Property data type
VALUELENGTH	DS	F	Length in bytes of the VALUE area
VALUE	DS	CL(n)	Area to contain the property value
DATALENGTH	DS	F	Length of the property value
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## MQMHBUF-Conversion du descripteur de message en mémoire tampon

L'appel MQMHBUF convertit un descripteur de message en mémoire tampon et est l'inverse de l'appel MQBUFMH.

### Syntaxe

```

MQMHBUF (Hconn, Hmsg, MsgHBufOpts, Nom, MsgDesc, BufferLength, Buffer, DataLength, CompCode, Motif)

```

## Paramètres

### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* doit correspondre au descripteur de connexion qui a été utilisé pour créer le descripteur de message spécifié dans le paramètre **Hmsg**.

Si le descripteur de message a été créé à l'aide de MQHC\_UNASSOCIATED\_HCONN, une connexion valide doit être établie sur l'unité d'exécution qui supprime le descripteur de message. Si aucune connexion valide n'est établie, l'appel échoue avec MQRC\_CONNECTION\_BROKEN.

### Msg

Type: MQHMSG-entrée

Il s'agit du descripteur de message pour lequel une mémoire tampon est requise. La valeur a été renvoyée par un appel MQCRTMH précédent.

### MsgHBufOpts

Type: MQMHBO-entrée

La structure MQMHBO permet aux applications de spécifier des options qui contrôlent la façon dont les mémoires tampon sont produites à partir des descripteurs de message.

Pour plus d'informations, voir [«MQMHBO-Options de traitement des messages vers la mémoire tampon»](#), à la page 499.

### Nom

Type: MQCHARV-entrée

Nom de la ou des propriétés à placer dans la mémoire tampon.

Si aucune propriété correspondant au nom ne peut être trouvée, l'appel échoue avec MQRC\_PROPERTY\_NOT\_AVAILABLE.

Vous pouvez utiliser un caractère générique pour placer plusieurs propriétés dans la mémoire tampon. Pour ce faire, utilisez le caractère générique '%' à la fin du nom de la propriété. Ce caractère générique correspond à zéro ou plusieurs caractères, y compris le caractère '?'.

Dans le langage de programmation C, les variables macro suivantes sont définies pour l'interrogation de toutes les propriétés et de toutes les propriétés qui commencent par 'usr':

#### **MQPROP\_INQUIRE\_ALL**

Placer toutes les propriétés du message dans la mémoire tampon

#### **MQPROP\_INQUIRE\_ALL\_USR**

Placez toutes les propriétés du message qui commencent par les caractères 'usr:' dans la mémoire tampon.

Pour plus d'informations sur l'utilisation des noms de propriété, voir [Noms de propriété](#) et [Restrictions relatives aux noms de propriété](#).

### MsgDesc

Type : MQMD - entrée/sortie

La structure *MsgDesc* décrit le contenu de la zone tampon.

Dans la sortie, les zones *Encoding*, *CodedCharSetId* et *Format* sont définies pour décrire correctement le codage, l'identificateur de jeu de caractères et le format des données dans la zone tampon, tels qu'ils sont écrits par l'appel.

Les données de cette structure se trouvent dans le jeu de caractères et le codage de l'application.

### BufferLength

Type : MQLONG - entrée

*BufferLength* est la longueur de la zone tampon, en octets.

## Mémoire tampon

Type : MQBYTEExBufferLength - sortie

*Buffer* définit la zone devant contenir les propriétés de message. Vous devez aligner la mémoire tampon sur une limite de 4 octets.

Si *BufferLength* est inférieur à la longueur requise pour stocker les propriétés dans *Buffer*, MQMHBUF échoue avec MQRC\_PROPERTY\_VALUE\_TOO\_BIG.

Le contenu de la mémoire tampon peut changer même si l'appel échoue.

## DataLength

Type : MQLONG - sortie

*DataLength* est la longueur, en octets, des propriétés renvoyées dans la mémoire tampon. Si la valeur est zéro, aucune propriété ne correspond à la valeur indiquée dans *Name* et l'appel échoue avec le code anomalie MQRC\_PROPERTY\_NOT\_AVAILABLE.

Si *BufferLength* est inférieur à la longueur requise pour stocker les propriétés dans la mémoire tampon, l'appel MQMHBUF échoue avec MQRC\_PROPERTY\_VALUE\_TOO\_BIG, mais une valeur est toujours entrée dans *DataLength*. Cela permet à l'application de déterminer la taille de la mémoire tampon requise pour les propriétés, puis d'émettre à nouveau l'appel avec le *BufferLength* requis.

## CompCode

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

### MQCC\_OK

Achèvement réussi.

### MQCC\_FAILED

Echec de l'appel.

## Motif

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

### MQRC\_NONE

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

### MQRC\_ADAPTER\_NOT\_AVAILABLE

(2204, X'089C') Carte non disponible.

### MQRC\_ADAPTER\_SERV\_LOAD\_ERROR

(2130, X'852') Impossible de charger le module de service d'adaptateur.

### MQRC\_ASID\_MISMATCH

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

### MQRC\_MHBO\_ERREUR

(2501, X'095C') Le descripteur de message vers la structure des options de mémoire tampon n'est pas valide.

### MQRC\_BUFFER\_ERROR

(2004, X'07D4') Paramètre de mémoire tampon non valide.

### MQRC\_BUFFER\_LENGTH\_ERROR

(2005, X'07D5') Paramètre de longueur de mémoire tampon non valide.

### MQRC\_CALL\_IN\_PROGRESS

(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

### MQRC\_CONNECTION\_BROKEN

(2009, X'07D9') La connexion au gestionnaire de files d'attente a été perdue.

**MQRC\_DATA\_LENGTH\_ERROR**

(2010, X'07DA') Paramètre de longueur de données non valide.

**ERREUR MQRC\_HMSG\_ERROR**

(2460, X'099C') Descripteur de message non valide.

**MQRC\_MD\_ERROR**

(2026, X'07EA') Descripteur de message non valide.

**MQRC\_MSG\_HANDLE\_IN\_USE**

(2499, X'09C3') Descripteur de message déjà utilisé.

**MQRC\_OPTIONS\_ERROR**

(2046, X'07FE') Options non valides ou non cohérentes.

**MQRC\_PROPERTY\_NAME\_ERROR**

(2442, X'098A') Le nom de propriété n'est pas valide.

**MQRC\_PROPERTY\_NOT\_AVAILABLE**

(2471, X'09A7') Propriété non disponible.

**MQRC\_PROPERTY\_VALEUR\_TOO\_BIG**

(2469, X'09A5') La valeur BufferLength est trop petite pour contenir les propriétés spécifiées.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Appel C

```
MQMHBUF (Hconn, Hmsg, &MsgHBufOpts, &Name, &MsgDesc, BufferLength, Buffer,
         &DataLength, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN Hconn;          /* Connection handle */
MQHMSG  Hmsg;           /* Message handle */
MQMHBO  MsgHBufOpts;   /* Options that control the action of MQMHBUF */
MQCHARV Name;          /* Property name */
MQMD    MsgDesc;       /* Message descriptor */
MQLONG  BufferLength;   /* Length in bytes of the Buffer area */
MQBYTE  Buffer[n];     /* Area to contain the properties */
MQLONG  DataLength;    /* Length of the properties */
MQLONG  CompCode;     /* Completion code */
MQLONG  Reason;       /* Reason code qualifying CompCode */
```

## Remarques sur l'utilisation

MQMHBUF convertit un descripteur de message en mémoire tampon.

Vous pouvez l'utiliser avec un exit d'API MQGET pour accéder à certaines propriétés, à l'aide des API de propriété de message, puis les transmettre dans une mémoire tampon à une application conçue pour utiliser des en-têtes MQRFH2 plutôt que des descripteurs de message.

Cet appel est l'inverse de l'appel MQBUFMH, que vous pouvez utiliser pour analyser les propriétés de message d'une mémoire tampon dans un descripteur de message.

## Appel COBOL

```
CALL 'MQMHBUF' USING HCONN, HMSG, MSGHBUFOPTS, NAME, MSGDESC,
                   BUFFERLENGTH, BUFFER, DATALENGTH, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```

** Connection handle
01 HCONN          PIC S9(9) BINARY.
** Message handle
01 HMSG          PIC S9(18) BINARY.
** Options that control the action of MQMHBUF
01 MSGHBUFOPTS.
   COPY CMQMHBV.
** Property name
01 NAME
   COPY CMQCHRVA.
** Message descriptor
01 MSGDESC
   COPY CMQMDV.
** Length in bytes of the Buffer area */
01 BUFFERLENGTH PIC S9(9) BINARY.
** Area to contain the properties
01 BUFFER        PIC X(n).
** Length of the properties
01 DATALENGTH  PIC S9(9) BINARY.
** Completion code
01 COMPCODE     PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON       PIC S9(9) BINARY.

```

## Appel PL/I

```

call MQMHBUF (Hconn, Hmsg, MsgHBufOpts, Name, MsgDesc, BufferLength, Buffer,
DataLength, CompCode, Reason);

```

Déclarez les paramètres comme suit :

```

dcl Hconn          fixed bin(31); /* Connection handle */
dcl Hmsg           fixed bin(63); /* Message handle */
dcl MsgHBufOpts   like MQMHBO;   /* Options that control the action of MQMHBUF */
dcl Name          like MQCHARV;  /* Property name */
dcl MsgDesc       like MQMD;     /* Message descriptor */
dcl BufferLength   fixed bin(31); /* Length in bytes of the Buffer area */
dcl Buffer         char(n);       /* Area to contain the properties */
dcl DataLength    fixed bin(31); /* Length of the properties */
dcl CompCode      fixed bin(31); /* Completion code */
dcl Reason        fixed bin(31); /* Reason code qualifying CompCode */

```

## Appel d'assembleur de haut niveau

```

CALL MQMHBUF, (HCONN,HMSG,MSGHBUFOPTS,NAME,MSGDESC,BUFFERLENGTH,
BUFFER,DATALENGTH,COMPCODE,REASON)

```

Déclarez les paramètres comme suit :

HCONN	DS	F	Connection handle
HMSG	DS	D	Message handle
MSGHBUFOPTS	CMQMHBV	,	Options that control the action of MQMHBUF
NAME	CMQCHRVA	,	Property name
MSGDESC	CMQMDA	,	Message descriptor
BUFFERLENGTH	DS	F	Length in bytes of the BUFFER area
BUFFER	DS	CL(n)	Area to contain the properties
DATALENGTH	DS	F	Length of the properties
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## MQOPEN-Objet ouvert

L'appel MQOPEN établit l'accès à un objet.

Les types d'objet suivants sont valides:

- File d'attente (y compris les listes de distribution)
- Liste de noms
- Définition de processus
- Gestionnaire de files d'attente
- Topic

## Syntaxe


MQOPEN (*Hconn*, *ObjDesc*, *Options*, *Hobj*, *CompCode*, *Raison*)

## Paramètres

### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de Hconn a été renvoyée par un appel MQCONN ou MQCONNX précédent.

 Sur les applications z/OS for CICS, l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour *Hconn*:

### MQHC\_DEF\_HCONN

Descripteur de connexion par défaut.

### ObjDesc

Type: MQOD-entrée/sortie

Il s'agit d'une structure qui identifie l'objet à ouvrir. Pour plus de détails, voir «[MQOD-Descripteur d'objet](#)», à la page 501.

Si la zone `ObjectName` du paramètre **ObjDesc** correspond au nom d'une file d'attente modèle, une file d'attente locale dynamique est créée avec les attributs de la file d'attente modèle ; cela se produit quelles que soient les options que vous spécifiez dans le paramètre **Options**. Les opérations suivantes utilisant le `Hobj` renvoyé par l'appel MQOPEN sont effectuées sur la nouvelle file d'attente dynamique et non sur la file d'attente modèle. Cela est vrai même pour les appels MQINQ et MQSET. Le nom de la file d'attente modèle dans le paramètre **ObjDesc** est remplacé par le nom de la file d'attente dynamique créée. Le type de la file d'attente dynamique est déterminé par la valeur de l'attribut **DefinitionType** de la file d'attente modèle (voir «[Attributs des files d'attente](#)», à la page 875). Pour plus d'informations sur les options de fermeture applicables aux files d'attente dynamiques, voir la description de l'appel MQCLOSE.

### Options

Type : MQLONG - entrée

Vous devez spécifier au moins l'une des options suivantes:

- MQOO\_BROWSE
- MQOO\_INPUT\_\* (un seul de ces éléments)
- MQOO\_INTERROGATION
- MQOO\_SORTIE
- MQOO\_SET
- MQOO\_BIND\_\* (un seul de ces éléments)

Consultez le tableau suivant pour plus de détails sur ces options ; d'autres options peuvent être spécifiées selon les besoins. Pour spécifier plusieurs options, ajoutez les valeurs ensemble (n'ajoutez pas la même constante plusieurs fois) ou combinez les valeurs à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations bit). Les combinaisons qui ne sont pas valides sont notées ; toutes les autres combinaisons sont valides. Seules les options applicables au type d'objet spécifié par `ObjDesc` sont autorisées.

Tableau 553. Options MQOPEN valides pour les files d'attente et les rubriques

Option	Alias <sup>1</sup>	Local et modèle	Eloigné	Cluster non local	Liste de distribution	Topic
<u>MQOO_INPUT_AS_Q_DEF</u>	Oui	Oui	Non	Non	Non	Non
<u>MQOO_INPUT_SHARED</u>	Oui	Oui	Non	Non	Non	Non
<u>MQOO_INPUT_EXCLUSIVE</u>	Oui	Oui	Non	Non	Non	Non
<u>MQOO_SORTIE</u>	Oui	Oui	Oui	Oui	Oui	Oui
<u>MQOO_BROWSE</u>	Oui	Oui	Non	Non	Non	Non
<u>MQOO_CO_OP</u>	Oui	Oui	Non	Non	Non	Non
<u>MQOO_INTERROGATION</u>	Oui	Oui	<u>2</u>	Oui	Non	Non
<u>MQOO_SET</u>	Oui	Oui	<u>2</u>	Non	Non	Non
<u>MQOO_BIND_ON_OPEN</u> <sup>3</sup>	Oui	Oui	Oui	Oui	Oui	Non
<u>MQOO_BIND_NOT_FIXED</u> <sup>3</sup>	Oui	Oui	Oui	Oui	Oui	Non
<u>MQOO_BIND_ON_GROUP</u> <sup>3</sup>	Oui	Oui	Oui	Oui	Oui	Non
<u>MQOO_BIND_AS_Q_DEF</u> <sup>3</sup>	Oui	Oui	Oui	Oui	Oui	Non
<u>MQOO_SAVE_ALL_CONTEXT</u>	Oui	Oui	Non	Non	Non	Non
<u>MQOO_PASS_IDENTITY_CONTEXT</u>	Oui	Oui	Oui	Oui	Oui	<u>4</u>
<u>MQOO_PASS_ALL_CONTEXT</u>	Oui	Oui	Oui	Oui	Oui	Oui
<u>MQOO_SET_IDENTITY_CONTEXT</u>	Oui	Oui	Oui	Oui	Oui	<u>4</u>
<u>MQOO_SET_ALL_CONTEXT</u>	Oui	Oui	Oui	Oui	Oui	Oui
<u>MQOO_NO_READ_AHEAD</u>	Oui	Oui	Non	Non	Non	Non
<u>MQOO_READ_AHEAD</u>	Oui	Oui	Non	Non	Non	Non
<u>MQOO_READ_AHEAD_AS_Q_DEF</u>	Oui	Oui	Non	Non	Non	Non
<u>MQOO_ALTERNATE_USER_AUTHORITY</u>	Oui	Oui	Oui	Oui	Oui	Oui
<u>MQOO_FAIL_IF QUIESCING</u>	Oui	Oui	Oui	Oui	Oui	Oui
<u>MQOO_RESOLVE_LOCAL_Q</u>	Oui	Oui	Oui	Oui	Non	Non
<u>MQOO_RESOLVE_LOCAL_TOPIC</u>	Non	Non	Non	Non	Non	Oui
<u>MQOO_NO_MULTICAST</u>	Non	Non	Non	Non	Non	Oui

**Remarques :**

1. La validité des options pour les alias dépend de la validité de l'option pour la file d'attente dans laquelle l'alias est résolu.
2. Cette option est valide uniquement pour la définition locale d'une file d'attente éloignée.
3. Cette option peut être spécifiée pour n'importe quel type de file d'attente, mais elle est ignorée si la file d'attente n'est pas une file d'attente de cluster. Toutefois, l'attribut de file d'attente **DefBind** remplace la file d'attente de base même si la file d'attente alias ne se trouve pas dans un cluster.
4. Ces attributs peuvent être utilisés avec une rubrique, mais affectent uniquement le contexte défini pour le message conservé, et non les zones de contexte envoyées à un abonné.

**Options d'accès:** Les options suivantes contrôlent le type d'opérations pouvant être effectuées sur l'objet:

**MQOO\_INPUT\_AS\_Q\_DEF**

Ouvrez la file d'attente pour obtenir les messages en utilisant la valeur par défaut définie par la file d'attente.



La file d'attente est ouverte pour être utilisée avec les appels MQGET suivants. Le type d'accès est partagé ou exclusif, en fonction de la valeur de l'attribut de file d'attente **DefInputOpenOption** ; voir «Attributs des files d'attente», à la page 875 pour plus de détails.

Cette option est valide uniquement pour les files d'attente locales, d'alias et modèles ; elle n'est pas valide pour les files d'attente éloignées, les listes de distribution et les objets qui ne sont pas des files d'attente.

### **MQOO\_INPUT\_SHARED**

Ouvrez la file d'attente pour obtenir les messages avec accès partagé.

La file d'attente est ouverte pour être utilisée avec les appels MQGET suivants. L'appel peut aboutir si la file d'attente est actuellement ouverte par cette application ou par une autre application avec MQOO\_INPUT\_SHARED, mais échoue avec le code anomalie MQRC\_OBJECT\_IN\_USE si la file d'attente est actuellement ouverte avec MQOO\_INPUT\_EXCLUSIVE.

Cette option est valide uniquement pour les files d'attente locales, d'alias et modèles ; elle n'est pas valide pour les files d'attente éloignées, les listes de distribution et les objets qui ne sont pas des files d'attente.

### **MQOO\_INPUT\_EXCLUSIVE**

Ouvrez la file d'attente pour obtenir les messages avec un accès exclusif.

La file d'attente est ouverte pour être utilisée avec les appels MQGET suivants. L'appel échoue avec le code anomalie MQRC\_OBJECT\_IN\_USE si la file d'attente est actuellement ouverte par cette application ou par une autre application pour l'entrée d'un type quelconque (MQOO\_INPUT\_SHARED ou MQOO\_INPUT\_EXCLUSIVE).

Cette option est valide uniquement pour les files d'attente locales, d'alias et modèles ; elle n'est pas valide pour les files d'attente éloignées, les listes de distribution et les objets qui ne sont pas des files d'attente.

### **MQOO\_SORTIE**

Ouvrez une file d'attente pour insérer des messages, ou une rubrique ou une chaîne de rubrique pour publier des messages.

La file d'attente ou la rubrique est ouverte pour être utilisée avec les appels MQPUT suivants.

Un appel MQOPEN avec cette option peut aboutir même si l'attribut de file d'attente **InhibitPut** est défini sur MQQA\_PUT\_INHIBÉ (bien que les appels MQPUT suivants échouent alors que l'attribut est défini sur cette valeur).

Cette option est valide pour tous les types de file d'attente, y compris les listes de distribution et les rubriques.

Les remarques suivantes s'appliquent à ces options:

- Une seule de ces options peut être spécifiée.
- Un appel MQOPEN avec l'une de ces options peut aboutir même si l'attribut de file d'attente **InhibitGet** est défini sur MQQA\_GET\_INHIBÉ (bien que les appels MQGET suivants échouent alors que l'attribut est défini sur cette valeur).
- Si la file d'attente est définie comme n'étant pas partageable (c'est-à-dire que l'attribut de file d'attente **Shareability** a la valeur MQQA\_NOT\_SHAREABLE), les tentatives d'ouverture de la file d'attente pour un accès partagé sont traitées comme des tentatives d'ouverture de la file d'attente avec un accès exclusif.
- Si une file d'attente alias est ouverte avec l'une de ces options, le test d'utilisation exclusive (ou de l'utilisation exclusive d'une autre application) est effectué sur la file d'attente de base dans laquelle l'alias est résolu.
- Ces options ne sont pas valides si **ObjectQMgrName** est le nom d'un alias de gestionnaire de files d'attente ; cela est vrai même si la valeur de l'attribut **RemoteQMgrName** dans la définition locale

d'une file d'attente éloignée utilisée pour l'alias de gestionnaire de files d'attente est le nom du gestionnaire de files d'attente local.

### **MQOO\_BROWSE**

Ouvrez la file d'attente pour parcourir les messages.

La file d'attente est ouverte pour être utilisée avec les appels MQGET suivants avec l'une des options suivantes:

- MQGMO\_BROWSE\_FIRST
- MQGMO\_BROWSE\_NEXT
- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR

Ceci est autorisé même si la file d'attente est actuellement ouverte pour MQOO\_INPUT\_EXCLUSIVE. Un appel MQOPEN avec l'option MQOO\_BROWSE établit un curseur de navigation et le positionne logiquement avant le premier message de la file d'attente ; voir [Zone MQGMO-Options](#) pour plus d'informations.

Cette option est valide uniquement pour les files d'attente locales, d'alias et modèles ; elle n'est pas valide pour les files d'attente éloignées, les listes de distribution et les objets qui ne sont pas des files d'attente. Elle n'est pas non plus valide si `ObjectQMgrName` est le nom d'un alias de gestionnaire de files d'attente ; cela est vrai même si la valeur de l'attribut **RemoteQMgrName** dans la définition locale d'une file d'attente éloignée utilisée pour l'alias de gestionnaire de files d'attente est le nom du gestionnaire de files d'attente local.

### **MQOO\_CO\_OP**

Ouvert en tant que membre coopérant de l'ensemble de poignées.

Cette option est valide uniquement avec l'option MQOO\_BROWSE. S'il est spécifié sans MQOO\_BROWSE, MQOPEN renvoie MQRC\_OPTIONS\_ERROR.

Le descripteur renvoyé est considéré comme étant membre d'un ensemble de descripteurs coopérant pour les appels MQGET suivants avec l'une des options suivantes:

- MQGMO\_MARK\_BROWSE\_CO\_OP
- MQGMO\_UNMARKED\_BROWSE\_MSG
- MQGMO\_UNMARK\_BROWSE\_CO\_OP

Cette option est valide uniquement pour les files d'attente locales, d'alias et modèles ; elle n'est pas valide pour les files d'attente éloignées, les listes de distribution et les objets qui ne sont pas des files d'attente.

### **MQOO\_INTERROGATION**

Ouvrez l'objet pour interroger les attributs.

La file d'attente, la liste de noms, la définition de processus ou le gestionnaire de files d'attente est ouvert pour être utilisé avec les appels MQINQ suivants.

Cette option est valide pour tous les types d'objet autres que les listes de distribution. Elle n'est pas valide si `ObjectQMgrName` est le nom d'un alias de gestionnaire de files d'attente ; ceci est vrai même si la valeur de l'attribut **RemoteQMgrName** dans la définition locale d'une file d'attente éloignée utilisée pour l'alias de gestionnaire de files d'attente est le nom du gestionnaire de files d'attente local.

### **MQOO\_SET**

Ouvrez la file d'attente pour définir les attributs.

La file d'attente est ouverte pour être utilisée avec les appels MQSET suivants.

Cette option est valide pour tous les types de file d'attente autres que les listes de distribution. Elle n'est pas valide si `ObjectQMgrName` est le nom d'une définition locale d'une file d'attente éloignée ; cela est vrai même si la valeur de l'attribut **RemoteQMgrName** dans la définition locale d'une file d'attente éloignée utilisée pour l'attribution d'alias au gestionnaire de files d'attente est le nom du gestionnaire de files d'attente local.

**Options de liaison:** Les options suivantes s'appliquent lorsque l'objet ouvert est une file d'attente de cluster ; ces options contrôlent la liaison du descripteur de file d'attente à une instance de la file d'attente de cluster:

#### **MQOO\_BIND\_ON\_OPEN**

Le gestionnaire de files d'attente local lie l'identificateur de file d'attente à une instance de la file d'attente de destination lorsque la file d'attente est ouverte. Par conséquent, tous les messages insérés à l'aide de cet identificateur sont envoyés à la même instance de la file d'attente de destination et par la même route.

Cette option est valide uniquement pour les files d'attente et n'affecte que les files d'attente de cluster. Si cette option est spécifiée pour une file d'attente qui n'est pas une file d'attente de cluster, elle est ignorée.

#### **MQOO\_BIND\_NOT\_FIXED**

Cette opération arrête le gestionnaire de files d'attente local qui lie le descripteur de file d'attente à une instance de la file d'attente de destination. Par conséquent, les appels MQPUT successifs utilisant ce descripteur envoient les messages à différentes instances de la file d'attente de destination ou à la même instance, mais par des routes différentes. Elle permet également à l'instance sélectionnée d'être modifiée ultérieurement par le gestionnaire de files d'attente local, par un gestionnaire de files d'attente éloignées ou par un agent MCA, en fonction des conditions du réseau.

**Remarque :** Les applications client et serveur qui doivent échanger une série de messages pour terminer une transaction ne doivent pas utiliser MQOO\_BIND\_NOT\_FIXED (ou MQOO\_BIND\_AS\_Q\_DEF lorsque DefBind a la valeur MQBND\_BIND\_NOT\_FIXED), car les messages successifs de la série peuvent être envoyés à différentes instances de l'application serveur.

Si MQOO\_BROWSE ou l'une des options MQOO\_INPUT\_\* est spécifiée pour une file d'attente de cluster, le gestionnaire de files d'attente est forcé de sélectionner l'instance locale de la file d'attente de cluster. Par conséquent, la liaison du descripteur de file d'attente est fixe, même si MQOO\_BIND\_NOT\_FIXED est spécifié.

Si MQOO\_INQUIRE est spécifié avec MQOO\_BIND\_NOT\_FIXED, les appels MQINQ successifs utilisant ce descripteur peuvent interroger différentes instances de la file d'attente de cluster, bien que toutes les instances aient généralement les mêmes valeurs d'attribut.

MQOO\_BIND\_NOT\_FIXED est valide uniquement pour les files d'attente et affecte uniquement les files d'attente de cluster. Si cette option est spécifiée pour une file d'attente qui n'est pas une file d'attente de cluster, elle est ignorée.

#### **MQOO\_BIND\_ON GROUPE**

Permet à une application de demander qu'un groupe de messages soit alloué à la même instance de destination.

Cette option est valide uniquement pour les files d'attente et n'affecte que les files d'attente de cluster. Si cette option est spécifiée pour une file d'attente qui n'est pas une file d'attente de cluster, elle est ignorée.

#### **MQOO\_BIND\_AS\_Q\_DEF**

Le gestionnaire de files d'attente local lie le descripteur de file d'attente de la manière définie par l'attribut de file d'attente **DefBind** . La valeur de cet attribut est MQBND\_BIND\_ON\_OPEN, MQBND\_BIND\_NOT\_FIXED ou MQBND\_BIND\_ON\_GROUP.

MQOO\_BIND\_AS\_Q\_DEF est la valeur par défaut lorsque MQOO\_BIND\_ON\_OPEN, MQOO\_BIND\_NOT\_FIXED ou MQOO\_BIND\_ON\_GROUP n'est pas spécifié.

MQOO\_BIND\_AS\_Q\_DEF aide à la documentation du programme. Il n'est pas prévu que cette option soit utilisée avec l'une des deux autres options de liaison, mais comme sa valeur est égale à zéro, cette utilisation ne peut pas être détectée.

**Options de contexte:** Les options suivantes contrôlent le traitement du contexte de message:

## **MQOO\_SAVE\_ALL\_CONTEXT**

Les informations de contexte sont associées à cet identificateur de file d'attente. Ces informations sont définies à partir du contexte de tout message extrait à l'aide de cet identificateur. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

Ces informations de contexte peuvent être transmises à un message qui est ensuite inséré dans une file d'attente à l'aide des appels MQPUT ou MQPUT1 . Voir les options MQPMO\_PASS\_IDENTITY\_CONTEXT et MQPMO\_PASS\_ALL\_CONTEXT décrites dans «[MQPMO-Options de message d'insertion](#)», à la page 522.

Tant qu'un message n'a pas été extrait, le contexte ne peut pas être transmis à un message en cours d'insertion dans une file d'attente.

Les informations de contexte d'un message extrait à l'aide de l'une des options de navigation MQGMO\_BROWSE\_\* ne sont pas sauvegardées (bien que les zones de contexte du paramètre **MsgDesc** soient définies après une navigation).

Cette option est valide uniquement pour les files d'attente locales, d'alias et modèles ; elle n'est pas valide pour les files d'attente éloignées, les listes de distribution et les objets qui ne sont pas des files d'attente. L'une des options MQOO\_INPUT\_\* doit être spécifiée.

## **MQOO\_PASS\_IDENTITY\_CONTEXT**

Cela permet à l'option MQPMO\_PASS\_IDENTITY\_CONTEXT d'être spécifiée dans le paramètre **PutMsgOpts** lorsqu'un message est inséré dans une file d'attente. Cette option fournit au message les informations de contexte d'identité d'une file d'attente d'entrée qui a été ouverte avec l'option MQOO\_SAVE\_ALL\_CONTEXT. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

L'option MQOO\_OUTPUT doit être spécifiée.

Cette option est valide pour tous les types de file d'attente, y compris les listes de distribution.

## **MQOO\_PASS\_ALL\_CONTEXT**

Cela permet à l'option MQPMO\_PASS\_ALL\_CONTEXT d'être spécifiée dans le paramètre **PutMsgOpts** lorsqu'un message est inséré dans une file d'attente ; cela donne au message les informations d'identité et de contexte d'origine à partir d'une file d'attente d'entrée qui a été ouverte avec l'option MQOO\_SAVE\_ALL\_CONTEXT. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

Cette option implique MQOO\_PASS\_IDENTITY\_CONTEXT, qui n'a donc pas besoin d'être spécifié. L'option MQOO\_OUTPUT doit être spécifiée.

Cette option est valide pour tous les types de file d'attente, y compris les listes de distribution.

## **MQOO\_SET\_IDENTITY\_CONTEXT**

Cela permet de spécifier l'option MQPMO\_SET\_IDENTITY\_CONTEXT dans le paramètre **PutMsgOpts** lorsqu'un message est inséré dans une file d'attente ; cela fournit au message les informations de contexte d'identité contenues dans le paramètre **MsgDesc** spécifié dans l'appel MQPUT ou MQPUT1 . Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

Cette option implique MQOO\_PASS\_IDENTITY\_CONTEXT, qui n'a donc pas besoin d'être spécifié. L'option MQOO\_OUTPUT doit être spécifiée.

Cette option est valide pour tous les types de file d'attente, y compris les listes de distribution.

## **MQOO\_SET\_ALL\_CONTEXT**

Cela permet à l'option MQPMO\_SET\_ALL\_CONTEXT d'être spécifiée dans le paramètre **PutMsgOpts** lorsqu'un message est inséré dans une file d'attente ; cela fournit au message les informations de contexte d'identité et d'origine contenues dans le paramètre **MsgDesc** spécifié dans l'appel MQPUT ou MQPUT1 . Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

Cette option implique les options suivantes, qui ne doivent donc pas être spécifiées:

- MQOO\_PASS\_IDENTITY\_CONTEXT
- MQOO\_PASS\_ALL\_CONTEXT
- MQOO\_SET\_IDENTITY\_CONTEXT

L'option MQOO\_OUTPUT doit être spécifiée.

Cette option est valide pour tous les types de file d'attente, y compris les listes de distribution.

### Options de lecture anticipée:

Lorsque vous appelez MQOPEN avec MQOO\_READ\_AHEAD, le client IBM MQ n'active la lecture anticipée que si certaines conditions sont remplies. Ces conditions sont les suivantes :

- L'application client doit être compilée et liée dans les bibliothèques client IBM MQ MQI des unités d'exécution.
- Le canal client doit utiliser le protocole TCP/IP.
- Le paramètre SharingConversations (SHARECNV) du canal doit avoir une valeur différente de zéro dans la définition de canal serveur et dans la définition de canal client.

Les options suivantes contrôlent si les messages non persistants sont envoyés au client avant qu'une application ne les demande. Les remarques suivantes s'appliquent aux options de lecture anticipée:

- Une seule de ces options peut être spécifiée.
- Ces options sont valides uniquement pour les files d'attente locales, d'alias et modèles. Ils ne sont pas valides pour les files d'attente éloignées, les listes de distribution, les rubriques ou les gestionnaires de files d'attente.
- Ces options ne sont applicables que lorsque l'une des options MQOO\_BROWSE, MQOO\_INPUT\_SHARED et MQOO\_INPUT\_EXCLUSIVE est également spécifiée, bien qu'il ne s'agisse pas d'une erreur de spécification de ces options avec MQOO\_INQUIRE ou MQOO\_SET.
- Si l'application n'est pas exécutée en tant que client IBM MQ , ces options sont ignorées.

### MQOO\_NO\_READ\_AHEAD

Les messages non persistants ne sont pas envoyés au client avant qu'une application ne les demande.

### MQOO\_READ\_AHEAD

Les messages non persistants sont envoyés au client avant qu'une application ne les demande.

### MQOO\_READ\_AHEAD\_AS\_Q\_DEF

Le comportement de lecture anticipée est déterminé par l'attribut de lecture anticipée par défaut de la file d'attente en cours d'ouverture. Il s'agit de la valeur par défaut.

**Autres options:** Les options suivantes contrôlent la vérification des autorisations, ce qui se produit lorsque le gestionnaire de files d'attente est mis au repos, s'il faut ou non résoudre le nom de la file d'attente locale et la multidiffusion:


### MQOO\_ALTERNATE\_USER\_AUTHORITY

La zone *AlternateUserId* du paramètre **ObjDesc** contient un identificateur utilisateur à utiliser pour valider cet appel MQOPEN. L'appel peut aboutir uniquement si ce *AlternateUserId* est autorisé à ouvrir l'objet avec les options d'accès spécifiées, que l'ID utilisateur sous lequel l'application est exécutée soit ou non autorisé à le faire. Cela ne s'applique pas aux options de contexte spécifiées, qui sont toujours vérifiées par rapport à l'identificateur utilisateur sous lequel l'application s'exécute.

Cette option est valide pour tous les types d'objet.

### MQOO\_FAIL\_IF QUIESCING

L'appel MQOPEN échoue si le gestionnaire de files d'attente est à l'état de mise au repos.

 Sous z/OS, pour une application CICS ou IMS , cette option force également l'appel MQOPEN à échouer si la connexion est à l'état de mise au repos.

Cette option est valide pour tous les types d'objet.

Pour plus d'informations sur les canaux client, voir [IBM MQ MQI clients](#).

### **MQOO\_RESOLVE\_LOCAL\_Q**

Remplissez ResolvedQName dans la structure MQOD avec le nom de la file d'attente locale qui a été ouverte. De même, le nom ResolvedQMgres est renseigné avec le nom du gestionnaire de files d'attente local qui héberge la file d'attente locale. Si la structure MQOD est inférieure à la version 3, MQOO\_RESOLVE\_LOCAL\_Q est ignoré et aucune erreur n'est renvoyée.

La file d'attente locale est toujours renvoyée lorsqu'une file d'attente locale, alias ou modèle est ouverte, mais ce n'est pas le cas lorsque, par exemple, une file d'attente éloignée ou une file d'attente de cluster non locale est ouverte sans l'option MQOO\_RESOLVE\_LOCAL\_Q ; les noms ResolvedQName et ResolvedQMgres sont renseignés avec les noms RemoteQName et RemoteQMgr trouvés dans la définition de file d'attente éloignée, ou de la même manière avec la file d'attente de cluster éloignée choisie.

Si vous spécifiez MQOO\_RESOLVE\_LOCAL\_Q lors de l'ouverture, par exemple, d'une file d'attente éloignée, ResolvedQName est la file d'attente de transmission dans laquelle les messages sont insérés. Le nom ResolvedQMgres est renseigné avec le nom du gestionnaire de files d'attente local hébergeant la file d'attente de transmission.

Si vous êtes autorisé à parcourir, entrer ou sortir dans une file d'attente, vous disposez des droits requis pour spécifier cet indicateur dans l'appel MQOPEN. Aucun droit spécial n'est requis.

Cette option est valide uniquement pour les files d'attente et les gestionnaires de files d'attente.

### **MQOO\_RESOLVE\_LOCAL\_TOPIC**

Renseignez ResolvedQName dans la structure MQOD en indiquant le nom de la rubrique d'administration ouverte.

### **MQOO\_NO\_MULTIDIFFUSION**

Les messages de publication ne sont pas envoyés à l'aide de la multidiffusion.

Cette option est valide uniquement avec l'option MQOO\_OUTPUT. S'il est spécifié sans MQOO\_OUTPUT, MQOPEN est renvoyé avec MQRC\_OPTIONS\_ERROR.

Cette option est valide uniquement pour une rubrique.

## **HOBJ**

Type: MQHOBJ-sortie

Ce descripteur représente l'accès qui a été établi à l'objet. Il doit être spécifié lors des appels IBM MQ suivants qui fonctionnent sur l'objet. Elle cesse d'être valide lorsque l'appel MQCLOSE est émis ou lorsque l'unité de traitement qui définit la portée de l'identificateur s'arrête.

La portée du descripteur d'objet renvoyé est identique à celle du descripteur de connexion spécifié dans l'appel. Pour plus d'informations sur la portée du descripteur, voir [MQCONN-Hconn parameter](#).

## **CompCode**

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

### **MQCC\_OK**

Achèvement réussi.

### **MQCC\_WARNING**

Avertissement (achèvement partiel).

### **MQCC\_FAILED**

Echec de l'appel.

## **Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_WARNING:

**MQRC\_MULTIPLE\_MOTIFS**

(2136, X'858') Plusieurs codes anomalie ont été renvoyés.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') Adaptateur non disponible.

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

**MQRC\_ALIAS\_BASE\_Q\_TYPE\_ERROR**

(2001, X'7D1') Le type de la file d'attente de base de l'alias n'est pas valide.

**MQRC\_API\_EXIT\_ERROR**

(2374, X'946') Echec de l'exit API.

**MQRC\_API\_EXIT\_LOAD\_ERROR**

(2183, X'887') Impossible de charger l'exit API.

**MQRC\_ASID\_MISMATCH**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

**MQRC\_CF\_NOT\_XX\_ENCODE\_CASE\_ONE disponible**

(2345, X' 929') Unité de couplage non disponible.

**Echec de MQRC\_CF\_STRUC\_AUTH\_FAILED**

(2348, X'92C') La vérification d'autorisation de la structure d'unité de couplage a échoué.

**MQRC\_CF\_STRUC\_ERREUR**

(2349, X'92D') Structure d'unité de couplage non valide.

**MQRC\_CF\_STRUC\_FAILED**

(2373, X'945') Echec de la structure de l'unité de couplage.

**MQRC\_CF\_STRUC\_IN\_USE**

(2346, X'92A') Structure de l'unité de couplage en cours d'utilisation.

**MQRC\_CF\_STRUC\_LIST\_HDR\_IN\_USE**

(2347, X'92B') En-tête de liste de la structure d'unité de couplage en cours d'utilisation.

**MQRC\_CICS\_WAIT\_FAILED**

(2140, X'85C') Demande d'attente rejetée par CICS.

**MQRC\_CLUSTER\_EXIT\_ERREUR**

(2266, X'8DA') L'exit de charge de travail de cluster a échoué.

**MQRC\_CLUSTER\_PUT\_INHIBÉ**

(2268, X'8DC') Appels d'insertion bloqués pour toutes les files d'attente du cluster.

**Erreur de résolution MQRC\_CLUSTER\_RESOLUTION\_ERROR**

(2189, X'88D') La résolution du nom de cluster a échoué.

**MQRC\_CLUSTER\_RESOURCE\_ERROR**

(2269, X'8DD') Erreur de ressource de cluster.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**MQRC\_CONNECTION\_NOT\_AUTHORIZED**

(2217, X'8A9') Non autorisé pour la connexion.

**MQRC\_CONNECTION QUIESCING**

(2202, X'89A') Connexion en cours de mise au repos.

**MQRC\_CONNECTION\_STOPPING**  
(2203, X'89B') Connexion en cours d'arrêt.

**MQRC\_DB2\_NOT\_AVAILABLE**  
(2342, X' 926') Sous-système Db2 non disponible.

**MQRC\_DEF\_XMIT\_Q\_TYPE\_ERREUR**  
(2198, X'896') La file d'attente de transmission par défaut n'est pas locale.

**MQRC\_DEF\_XMIT\_Q\_USAGE\_ERROR**  
(2199, X'897') Erreur d'utilisation de la file d'attente de transmission par défaut.

**MQRC\_DYNAMIC\_Q\_NAME\_ERROR**  
(2011, X'7DB') Nom de la file d'attente dynamique non valide.

**MQRC\_HANDLE\_NOT\_AVAILABLE**  
(2017, X'7E1') Plus de descripteurs disponibles.

**MQRC\_HCONN\_ERROR**  
(2018, X'7E2') Descripteur de connexion non valide.

**MQRC\_HOBJ\_ERROR**  
(2019, X'7E3') Descripteur d'objet non valide.

**MQRC\_MULTIPLE\_MOTIFS**  
(2136, X'858') Plusieurs codes anomalie ont été renvoyés.

**MQRC\_NAME\_IN\_USE**  
(2201, X'899') Nom utilisé.

**MQRC\_NAME\_NOT\_VALID\_FOR\_TYPE**  
(2194, X'892') Nom d'objet incorrect pour le type d'objet.

**MQRC\_NOT\_AUTHORIZED**  
(2035, X'7F3') Non autorisé pour l'accès.

**MQRC\_OBJECT\_ALREADY\_EXISTS**  
(2100, X'834') L'objet existe.

**MQRC\_OBJECT\_DAMAGED**  
(2101, X'835') Objet endommagé.

**MQRC\_OBJECT\_IN\_USE**  
(2042, X'7FA') Objet déjà ouvert avec des options en conflit.

**MQRC\_OBJET\_NIVEAU\_INCOMPATIBLE**  
(2360, X' 938') Niveau d'objet non compatible.

**MQRC\_OBJECT\_NAME\_ERROR**  
(2152, X'868') Nom d'objet incorrect.

**MQRC\_OBJECT\_NOT\_UNIQUE**  
(2343, X' 927') Objet non unique.

**MQRC\_OBJECT\_Q\_MGR\_NAME\_ERROR**  
(2153, X'869') Nom de gestionnaire de files d'attente d'objet incorrect.

**MQRC\_OBJECT\_RECORDS\_ERROR**  
(2155, X'86B') Enregistrements d'objet non valides.

**MQRC\_OBJECT\_STRING\_ERROR**  
(2441, X'0989') Zone Objectstring non valide

**MQRC\_OBJECT\_TYPE\_ERREUR**  
(2043, X'7FB') Type d'objet non valide.

**MQRC\_ERREUR D'OD\_DU**  
(2044, X'7FC') Structure de descripteur d'objet non valide.

**MQRC\_OPTION\_NOT\_VALID\_FOR\_TYPE**  
(2045, X'7FD') Option non valide pour le type d'objet.

**MQRC\_OPTIONS\_ERROR**  
(2046, X'7FE') Options incorrectes ou incohérentes.



**MQRC\_PAGESET\_ERROR**

(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.

**MQRC\_PAGESET\_FULL**

(2192, X'890') Support de stockage externe saturé.

**MQRC\_Q\_DELETED**

(2052, X'804') File d'attente supprimée.

**MQRC\_Q\_MGR\_NAME\_ERROR**

(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**

(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

**MQRC\_Q\_MGR QUIESCING**

(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**MQRC\_Q\_TYPE\_ERREUR**

(2057, X'809') Type de file d'attente non valide.

**MQRC\_RECS\_Présentateur-ERREUR**

(2154, X'86A') Nombre d'enregistrements présents non valide.

**MQRC\_REMOTE\_Q\_NAME\_ERROR**

(2184, X'888') Nom de file d'attente éloignée incorrect.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') Ressources système disponibles insuffisantes.

**MQRC\_RESPONSE\_RECORDS\_ERROR**

(2156, X'86C') Enregistrements de réponse non valides.

**MQRC\_SECURITY\_ERROR (ERREUR DE SECURITE MQ)**

(2063, X'80F') Une erreur de sécurité s'est produite.

**MQRC\_SELECTOR\_SYNTAX\_ERREUR**

2459 (X'099B') Un appel MQOPEN, MQPUT1 ou MQSUB a été émis mais une chaîne de sélection contenant une erreur de syntaxe a été spécifiée.

**MQRC\_STOPPED\_BY\_CLUSTER\_EXIT**

(2188, X'88C') Appel rejeté par l'exit de charge de travail du cluster.

**MQRC\_STORAGE\_MEDIUM\_FULL**

(2192, X'890') Support de stockage externe saturé.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') Mémoire disponible insuffisante.

**MQRC\_SUPPRESSED\_BY\_EXIT**

(2109, X'83D') Appel supprimé par le programme d'exit.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') Une erreur inattendue s'est produite.

**MQRC\_UNKNOWN\_ALIAS\_BASE\_Q**

(2082, X'822') File d'attente de base alias inconnue.

**MQRC\_UNKNOWN\_DEF\_XMIT\_Q**

(2197, X'895') File d'attente de transmission par défaut inconnue.

**MQRC\_UNKNOWN\_NOM\_OBJET**

(2085, X'825') Nom d'objet inconnu.

**MQRC\_UNKNOWN\_OBJET\_Q\_MGR**

(2086, X'826') Gestionnaire de files d'attente d'objets inconnu.

**MQRC\_UNKNOWN\_REMOTE\_Q\_MGR**

(2087, X'827') Gestionnaire de files d'attente éloignées inconnu.

**MQRC\_UNKNOWN\_XMIT\_Q**

(2196, X'894') File d'attente de transmission inconnue.

**MQRC\_WRONG\_CF\_NIVEAU**

(2366, X'93E') La structure de l'unité de couplage est de niveau incorrect.

**MQRC\_XMIT\_Q\_TYPE\_ERROR**

(2091, X'82B') File d'attente de transmission non locale.

**MQRC\_XMIT\_Q\_USAGE\_ERROR**

(2092, X'82C') File d'attente de transmission avec une utilisation incorrecte.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Remarques générales sur l'utilisation


1. L'objet ouvert est l'un des suivants:

- Une file d'attente pour:
  - Extraire ou parcourir les messages (à l'aide de l'appel MQGET)
  - Messages d'insertion (à l'aide de l'appel MQPUT)
  - Renseignez-vous sur les attributs de la file d'attente (à l'aide de l'appel MQINQ)
  - Définir les attributs de la file d'attente (à l'aide de l'appel MQSET)

Si la file d'attente nommée est une file d'attente modèle, une file d'attente locale dynamique est créée. Voir le paramètre **ObjDesc** décrit dans «MQOPEN-Objet ouvert», à la page 766.

Une liste de distribution est un type spécial d'objet file d'attente qui contient une liste de files d'attente. Il peut être ouvert pour insérer des messages, mais pas pour extraire ou parcourir des messages, ni pour interroger ou définir des attributs. Voir la note d'utilisation 8 pour plus de détails.

Une file d'attente QSGDISP (GROUP) est un type spécial de définition de file d'attente qui ne peut pas être utilisé avec les appels MQOPEN ou MQPUT1.

- Liste de noms permettant de connaître les noms des files d'attente de la liste (à l'aide de l'appel MQINQ).
  - Une définition de processus pour l'interrogation des attributs de processus (à l'aide de l'appel MQINQ).
  - Le gestionnaire de files d'attente pour demander les attributs du gestionnaire de files d'attente local (à l'aide de l'appel MQINQ).
  - Rubrique permettant de publier un message (à l'aide de l'appel MQPUT)
2. Une application peut ouvrir le même objet plusieurs fois. Un descripteur d'objet différent est renvoyé pour chaque ouverture. Chaque descripteur renvoyé peut être utilisé pour les fonctions pour lesquelles l'ouverture correspondante a été effectuée.
3. Si l'objet ouvert est une file d'attente autre qu'une file d'attente de cluster, la résolution de tous les noms dans le gestionnaire de files d'attente local est effectuée au moment de l'appel MQOPEN. Cet audit peut porter sur l'un ou plusieurs des éléments suivants :
- Résolution du nom d'une définition locale d'une file d'attente éloignée en nom du gestionnaire de files d'attente éloignées et nom sous lequel la file d'attente est connue sur le gestionnaire de files d'attente éloignées
  - Résolution du nom du gestionnaire de files d'attente éloignées en nom de file d'attente de transmission locale
  -  Sous z/OS uniquement, la résolution du nom du gestionnaire de files d'attente éloignées par le nom de la file d'attente de transmission partagée utilisée par l'agent de mise en file d'attente intra-groupe (s'applique uniquement si les gestionnaires de files d'attente locales et éloignées appartiennent au même groupe de partage de files d'attente)
  - Résolution d'alias sur le nom d'une file d'attente de base ou d'un objet de rubrique.

Cependant, sachez que les appels MQINQ ou MQSET ultérieurs pour le descripteur se rapportent uniquement au nom qui a été ouvert et non à l'objet résultant de la résolution de nom. Par exemple, si l'objet ouvert est un alias, les attributs renvoyés par l'appel MQINQ sont les attributs de l'alias, et non les attributs de la file d'attente de base ou un objet de rubrique auquel l'alias est résolu.

Si l'objet en cours d'ouverture est une file d'attente de cluster, la résolution de nom peut se produire au moment de l'appel MQOPEN ou être différée jusqu'à une date ultérieure. Le point de résolution est contrôlé par les options MQOO\_BIND\_\* spécifiées dans l'appel MQOPEN:

- MQOO\_BIND\_ON\_OPEN
- MQOO\_BIND\_NOT\_FIXED
- MQOO\_BIND\_AS\_Q\_DEF
- MQOO\_BIND\_ON\_GROUPE

Voir [Résolution de nom](#) pour plus d'informations sur la résolution de nom pour les files d'attente de cluster.

4. Un appel MQOPEN avec l'option MQOO\_BROWSE établit un curseur de navigation, à utiliser avec les appels MQGET qui spécifient le descripteur d'objet et l'une des options de navigation. Cela permet d'analyser la file d'attente sans modifier son contenu. Un message trouvé lors de la consultation peut être supprimé de la file d'attente à l'aide de l'option MQGMO\_MSG\_UNDER\_CURSOR.

Plusieurs curseurs de navigation peuvent être actifs pour une seule application en émettant plusieurs demandes MQOPEN pour la même file d'attente.

5. Les applications démarrées par un moniteur de déclenchement sont transmises avec le nom de la file d'attente associée à l'application lors de son démarrage. Ce nom de file d'attente peut être spécifié dans le paramètre **ObjDesc** pour ouvrir la file d'attente. Pour plus de détails, voir [«MQTMC2 -Message de déclenchement 2 \(format de caractère\)»](#), à la page 634.

## Options de lecture anticipée

Lorsque vous appelez MQOPEN avec MQOO\_READ\_AHEAD, le client IBM MQ n'active la lecture anticipée que si certaines conditions sont remplies. Ces conditions sont les suivantes :

- L'application client doit être compilée et liée dans les bibliothèques client IBM MQ MQI des unités d'exécution.
- Le canal client doit utiliser le protocole TCP/IP.
- Le paramètre SharingConversations (SHARECNV) du canal doit avoir une valeur différente de zéro dans la définition de canal serveur et dans la définition de canal client.

Les remarques suivantes s'appliquent à l'utilisation des options de lecture anticipée.

1. Les options de lecture anticipée s'appliquent uniquement lorsqu'une seule des options MQOO\_BROWSE, MQOO\_INPUT\_SHARED et MQOO\_INPUT\_EXCLUSIVE est également spécifiée. Une erreur n'est pas générée si des options de lecture anticipée sont spécifiées avec les options MQOO\_INQUIRE ou MQOO\_SET.
2. La lecture anticipée n'est pas activée lorsqu'elle est demandée si les options utilisées sur le premier appel MQGET ne sont pas prises en charge pour une utilisation avec la lecture anticipée. En outre, la lecture anticipée est désactivée lorsque le client se connecte à un gestionnaire de files d'attente qui ne prend pas en charge la lecture anticipée.
3. Si l'application n'est pas exécutée en tant que client IBM MQ, les options de lecture anticipée sont ignorées.

## Files d'attente de cluster

Les remarques suivantes s'appliquent à l'utilisation des files d'attente de cluster.

1. Lorsqu'une file d'attente de cluster est ouverte pour la première fois et que le gestionnaire de files d'attente local n'est pas un gestionnaire de files d'attente de référentiel complet, le gestionnaire de files d'attente local obtient des informations sur la file d'attente de cluster à partir d'un gestionnaire

de files d'attente de référentiel complet. Lorsque le réseau est occupé, le gestionnaire de files d'attente local peut mettre plusieurs secondes à recevoir les informations requises du gestionnaire de files d'attente de référentiel. Par conséquent, l'application qui émet l'appel MQOPEN peut avoir à attendre jusqu'à 10 secondes avant que le contrôle ne soit renvoyé à partir de l'appel MQOPEN. Si le gestionnaire de files d'attente local ne reçoit pas les informations nécessaires sur la file d'attente de cluster dans ce délai, l'appel échoue avec le code anomalie MQRC\_CLUSTER\_RESOLUTION\_ERROR.

2. Lorsqu'une file d'attente de cluster est ouverte et qu'il existe plusieurs instances de la file d'attente dans le cluster, l'instance ouverte dépend des options spécifiées dans l'appel MQOPEN:

- Si les options spécifiées incluent l'une des options suivantes:

- MQOO\_BROWSE
- MQOO\_INPUT\_AS\_Q\_DEF
- MQOO\_INPUT\_EXCLUSIVE
- MQOO\_INPUT\_SHARED
- MQOO\_SET

l'instance de la file d'attente de cluster ouverte doit être l'instance locale. S'il n'existe aucune instance locale de la file d'attente, l'appel MQOPEN échoue.

- Si les options spécifiées n'incluent aucune des options décrites précédemment, mais incluent l'une des options suivantes ou les deux:

- MQOO\_INTERROGATION
- MQOO\_SORTIE

l'instance ouverte est l'instance locale, s'il en existe une, et une instance distante dans le cas contraire (si vous utilisez les valeurs par défaut de CLWLUSEQ). L'instance choisie par le gestionnaire de files d'attente peut toutefois être modifiée par un exit de charge de travail de cluster (s'il en existe un).

3. S'il existe un abonnement pour la file d'attente, mais qu'elle n'est pas reconnue par un référentiel complet, l'objet n'est pas présent dans le cluster et l'appel échoue avec le code anomalie MQRC\_OBJECT\_NAME.

Pour plus d'informations sur les files d'attente de cluster, voir [Files d'attente de cluster](#).

## Listes de diffusion

Les remarques suivantes s'appliquent à l'utilisation des listes de distribution.

Les listes de distribution sont prises en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour IBM MQ MQI clients connecté à ces systèmes.

1. Les zones de la structure MQOD doivent être définies comme suit lors de l'ouverture d'une liste de distribution:

- Version doit être MQOD\_VERSION\_2 ou supérieur.
- ObjectType doit être MQOT\_Q.
- ObjectName doit être vide ou la chaîne null.
- ObjectQMgrName doit être vide ou la chaîne null.
- RecsPresent être supérieur à zéro.

- L'un des éléments `ObjectRecOffset` et `ObjectRecPtr` doit être égal à zéro et l'autre doit être différent de zéro.
- `ResponseRecOffset` et `ResponseRecPtr` ne peuvent pas être différents de zéro.
- Il doit y avoir des enregistrements d'objet `RecsPresent`, traités par `ObjectRecOffset` ou `ObjectRecPtr`. Les enregistrements d'objet doivent être définis sur les noms des files d'attente de destination à ouvrir.
- Si l'un des éléments `ResponseRecOffset` et `ResponseRecPtr` est différent de zéro, des enregistrements de réponse `RecsPresent` doivent être présents. Ils sont définis par le gestionnaire de files d'attente si l'appel se termine avec le code anomalie `MQRC_MULTIPLE_MOTIFS`.

Un `MQOD` version-2 peut également être utilisé pour ouvrir une file d'attente unique qui ne figure pas dans une liste de distribution, en veillant à ce que `RecsPresent` soit égal à zéro.

2. Seules les options d'ouverture suivantes sont valides dans le paramètre **Options** :

- `MQOO_SORTIE`
- `MQOO_PASS_*_CONTEXTE`
- `MQOO_SET_*_CONTEXTE`
- `MQOO_ALTERNATE_USER_AUTHORITY`
- `MQOO_FAIL_IF_QUIESCING`

3. Les files d'attente de destination de la liste de distribution peuvent être des files d'attente locales, des files d'attente alias ou des files d'attente éloignées, mais elles ne peuvent pas être des files d'attente modèles. Si une file d'attente modèle est spécifiée, l'ouverture de cette file d'attente échoue avec le code anomalie `MQRC_Q_TYPE_ERROR`. Toutefois, cela n'empêche pas l'ouverture d'autres files d'attente de la liste.

4. Les paramètres de code achèvement et de code anomalie sont définis comme suit:

- Si les opérations d'ouverture des files d'attente de la liste de distribution aboutissent ou échouent de la même manière, les paramètres de code achèvement et de code anomalie sont définis pour décrire le résultat commun. Les enregistrements de réponse `MQRR` (s'ils sont fournis par l'application) ne sont pas définis dans ce cas.

Par exemple, si chaque ouverture aboutit, le code achèvement est défini sur `MQCC_OK` et le code anomalie est défini sur `MQRC_NON`; si chaque ouverture échoue car aucune des files d'attente n'existe, les paramètres sont définis sur `MQCC_FAILED` et `MQRC_UNKNOWN_OBJECT_NAME`.

- Si les opérations d'ouverture des files d'attente de la liste de distribution n'aboutissent pas toutes ou échouent de la même manière:
  - Le paramètre de code achèvement est défini sur `MQCC_WARNING` si au moins une ouverture a abouti et sur `MQCC_FAILED` si toutes les opérations ont échoué.
  - Le paramètre de code anomalie est défini sur `MQRC_MULTIPLE_MOTIFS`.
  - Les enregistrements de réponse (s'ils sont fournis par l'application) sont définis sur les codes achèvement individuels et les codes raison des files d'attente de la liste de distribution.

5. Lorsqu'une liste de distribution a été ouverte avec succès, le descripteur `Hobj` renvoyé par l'appel peut être utilisé sur les appels `MQPUT` suivants pour placer des messages dans les files d'attente de la liste de distribution et sur un appel `MQCLOSE` pour abandonner l'accès à la liste de distribution. La seule option de fermeture valide pour une liste de distribution est `MQCO_NONE`.

L'appel `MQPUT1` peut également être utilisé pour placer un message dans une liste de distribution ; la structure `MQOD` définissant les files d'attente dans la liste est spécifiée en tant que paramètre sur cet appel.

6. Chaque destination correctement ouverte dans la liste de distribution compte comme un descripteur distinct lors de la vérification du dépassement du nombre maximal de descripteurs autorisé par l'application (voir l'attribut de gestionnaire de files d'attente **MaxHandles**). Cela est vrai même lorsque deux ou plusieurs des destinations de la liste de distribution sont résolues dans la même file d'attente physique. Si l'appel `MQOPEN` ou `MQPUT1` pour une liste de distribution entraîne le

dépassement du nombre de descripteurs utilisés par l'application `MaxHandles`, l'appel échoue avec le code anomalie `MQRC_HANDLE_NOT_AVAILABLE`.

7. Chaque destination ouverte avec succès a la valeur de son attribut **OpenOutputCount** incrémentée d'un. Si au moins deux des destinations de la liste de distribution sont résolues dans la même file d'attente physique, l'attribut **OpenOutputCount** de cette file d'attente est incrémenté du nombre de destinations de la liste de distribution qui sont résolues dans cette file d'attente.
8. Toute modification apportée aux définitions de file d'attente qui aurait entraîné la non-validité d'un descripteur si les files d'attente avaient été ouvertes individuellement (par exemple, une modification du chemin de résolution) n'entraîne pas la non-validité du descripteur de liste de distribution. Toutefois, elle entraîne un échec pour cette file d'attente particulière lorsque le descripteur de liste de distribution est utilisé lors d'un appel `MQPUT` ultérieur.
9. Une liste de distribution ne peut contenir qu'une seule destination.

## Files d'attente éloignées

Les remarques suivantes s'appliquent à l'utilisation des files d'attente éloignées.

Une file d'attente éloignée peut être spécifiée de deux manières dans le paramètre **ObjDesc** de cet appel.

- En spécifiant pour `ObjectName` le nom d'une définition locale de la file d'attente éloignée. Dans ce cas, `ObjectQMgrName` fait référence au gestionnaire de files d'attente local et peut être spécifié sous forme de blancs ou (dans le langage de programmation C) sous la forme d'une chaîne nulle.

La validation de sécurité effectuée par le gestionnaire de files d'attente local vérifie que l'utilisateur est autorisé à ouvrir la définition locale de la file d'attente éloignée.

- En spécifiant pour `ObjectName` le nom de la file d'attente éloignée, tel qu'il est connu du gestionnaire de files d'attente éloignées. Dans ce cas, `ObjectQMgrName` est le nom du gestionnaire de files d'attente éloignées.

La validation de sécurité effectuée par le gestionnaire de files d'attente local vérifie que l'utilisateur est autorisé à envoyer des messages à la file d'attente de transmission résultant du processus de résolution de nom.

Dans les deux cas:

- Aucun message n'est envoyé par le gestionnaire de files d'attente local au gestionnaire de files d'attente éloignées pour vérifier que l'utilisateur est autorisé à placer des messages dans la file d'attente.
- Lorsqu'un message arrive sur le gestionnaire de files d'attente éloignées, ce dernier peut le rejeter car l'utilisateur à l'origine du message n'est pas autorisé.

Pour plus d'informations, voir les zones `ObjectName` et `ObjectQMgrName` décrites dans [«MQOD-Descripteur d'objet»](#), à la page 501 .

## Objets

### Sécurité

Les remarques suivantes concernent les aspects de sécurité liés à l'utilisation de `MQOPEN`.

Le gestionnaire de files d'attente effectue des contrôles de sécurité lorsqu'un appel `MQOPEN` est émis, afin de vérifier que l'ID utilisateur sous lequel l'application s'exécute dispose du niveau de droits approprié avant que l'accès ne soit autorisé. La vérification des droits est effectuée sur le nom de l'objet en cours d'ouverture et non sur le ou les noms, ce qui se produit après la résolution d'un nom.

Si l'objet ouvert est une file d'attente alias qui pointe vers un objet de rubrique, le gestionnaire de files d'attente effectue un contrôle de sécurité sur le nom de la file d'attente alias, avant d'effectuer un contrôle de sécurité pour la rubrique comme si l'objet de rubrique avait été utilisé directement.

Si l'objet en cours d'ouverture est un objet de rubrique, que ce soit avec `ObjectName` seul ou à l'aide de `ObjectString` (avec ou sans base `ObjectName`), le gestionnaire de files d'attente effectue le contrôle de sécurité à l'aide de la chaîne de rubrique résultante, extraite de l'objet de rubrique spécifié dans `ObjectName`, et, si nécessaire, en le concaténant avec celui fourni dans `ObjectString`, puis en recherchant l'objet de rubrique le plus proche à ce point ou au-dessus de ce point dans l'arborescence de rubriques pour effectuer le contrôle de sécurité. Il se peut qu'il ne s'agit pas du même objet de rubrique que celui spécifié dans `ObjectName`.

Si l'objet ouvert est une file d'attente modèle, le gestionnaire de files d'attente effectue une vérification de sécurité complète par rapport au nom de la file d'attente modèle et au nom de la file d'attente dynamique créée. Si la file d'attente dynamique résultante est ouverte de manière explicite, une vérification supplémentaire de la sécurité des ressources est effectuée par rapport au nom de la file d'attente dynamique.

**z/OS** Sous z/OS, le gestionnaire de files d'attente effectue des contrôles de sécurité uniquement si la sécurité est activée. Pour plus d'informations sur la vérification de la sécurité, voir [Configuration de la sécurité sur z/OS](#).

## Attributs

Les remarques suivantes concernent les attributs.

Les attributs d'un objet peuvent changer lorsque l'objet est ouvert dans une application. Dans de nombreux cas, l'application ne le remarque pas, mais pour certains attributs, le gestionnaire de files d'attente marque l'identificateur comme n'étant plus valide. Ces attributs sont les suivants:

- Tout attribut qui affecte la résolution de nom de l'objet. Cela s'applique indépendamment des options ouvertes utilisées et inclut les éléments suivants:
  - Modification de l'attribut **BaseQName** d'une file d'attente alias ouverte.
  - Modification de l'attribut **TargetType** d'une file d'attente alias ouverte.
  - Modification des attributs de file d'attente **RemoteQName** ou **RemoteQMgrName**, pour tout descripteur ouvert pour cette file d'attente ou pour une file d'attente résolue via cette définition en tant qu'alias de gestionnaire de files d'attente.
  - Toute modification entraînant la résolution d'un descripteur actuellement ouvert pour une file d'attente éloignée dans une autre file d'attente de transmission ou l'échec de la résolution en une seule file d'attente. Par exemple, il peut s'agir des éléments suivants:
    - Modification de l'attribut **XmitQName** de la définition locale d'une file d'attente éloignée, que la définition soit utilisée pour une file d'attente ou pour un alias de gestionnaire de files d'attente.
    - **z/OS** Sous z/OS uniquement, une modification de la valeur de l'attribut de gestionnaire de files d'attente **IntraGroupqueuing** ou une modification de la définition de la file d'attente de transmission partagée (SYSTEM.QSG.TRANSMIT.QUEUE) utilisée par l'agent IGQ.

Il y a une exception à cela: la création d'une nouvelle file d'attente de transmission. Un descripteur qui aurait été résolu dans cette file d'attente s'il avait été présent lors de l'ouverture du descripteur, mais qui aurait été résolu dans la file d'attente de transmission par défaut, n'est pas rendu non valide.

- Modification de l'attribut de gestionnaire de files d'attente **DefXmitQName**. Dans ce cas, tous les descripteurs ouverts résolus dans la file d'attente précédemment nommée (qui ont été résolus uniquement parce qu'il s'agissait de la file d'attente de transmission par défaut) sont marqués comme non valides. Les descripteurs résolus dans cette file d'attente pour d'autres raisons ne sont pas affectés.
- L'attribut de file d'attente **Shareability**, s'il existe au moins deux descripteurs qui fournissent actuellement un accès `MQOO_INPUT_SHARED` pour cette file d'attente ou pour une file d'attente qui se résout en cette file d'attente. Si tel est le cas, tous les descripteurs ouverts pour cette file d'attente ou pour une file d'attente qui se résout en cette file d'attente sont marqués comme non valides, quelles que soient les options d'ouverture.

**z/OS** Sous z/OS, les descripteurs décrits précédemment sont marqués comme non valides si un ou plusieurs descripteurs fournissent actuellement un accès MQOO\_INPUT\_SHARED ou MQOO\_INPUT\_EXCLUSIVE à la file d'attente.

- L'attribut de file d'attente **Usage**, pour tous les descripteurs ouverts pour cette file d'attente, ou pour une file d'attente qui se résout en cette file d'attente, quelles que soient les options d'ouverture.

Lorsqu'un descripteur est marqué comme non valide, tous les appels ultérieurs (autres que MQCLOSE) utilisant ce descripteur échouent avec le code anomalie MQRC\_OBJECT\_CHANGED. L'application doit émettre un appel MQCLOSE (à l'aide du descripteur d'origine), puis rouvrir la file d'attente. Toutes les mises à jour non validées par rapport à l'ancien descripteur des appels réussis précédents peuvent toujours être validées ou annulées, comme requis par la logique de l'application.

Si la modification d'un attribut entraîne cette situation, utilisez une version forcée spéciale de l'appel.

## Appel C

```
MQOPEN (Hconn, &ObjDesc, Options, &Hobj, &CompCode,  
&Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN Hconn; /* Connection handle */  
MQOD ObjDesc; /* Object descriptor */  
MQLONG Options; /* Options that control the action of MQOPEN */  
MQHOBJ Hobj; /* Object handle */  
MQLONG CompCode; /* Completion code */  
MQLONG Reason; /* Reason code qualifying CompCode */
```

## Appel COBOL

```
CALL 'MQOPEN' USING HCONN, OBJDESC, OPTIONS, HOBJ, COMPCODE, REASON
```

Déclarez les paramètres comme suit :

```
** Connection handle  
01 HCONN PIC S9(9) BINARY.  
** Object descriptor  
01 OBJDESC.  
COPY CMQODV.  
** Options that control the action of MQOPEN  
01 OPTIONS PIC S9(9) BINARY.  
** Object handle  
01 HOBJ PIC S9(9) BINARY.  
** Completion code  
01 COMPCODE PIC S9(9) BINARY.  
** Reason code qualifying COMPCODE  
01 REASON PIC S9(9) BINARY.
```

## Appel PL/I

```
call MQOPEN (Hconn, ObjDesc, Options, Hobj, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
dcl Hconn fixed bin(31); /* Connection handle */  
dcl ObjDesc like MQOD; /* Object descriptor */  
dcl Options fixed bin(31); /* Options that control the action of  
MQOPEN */  
dcl Hobj fixed bin(31); /* Object handle */
```



```
dcl CompCode fixed bin(31); /* Completion code */
dcl Reason fixed bin(31); /* Reason code qualifying CompCode */
```

## Appel d'assembleur de haut niveau

```
CALL MQOPEN, (HCONN, OBJDESC, OPTIONS, HOBJ, COMPCODE, REASON)
```

Déclarez les paramètres comme suit :

HCONN	DS	F	Connection handle
OBJDESC	CMQODA	,	Object descriptor
OPTIONS	DS	F	Options that control the action of MQOPEN
HOBJ	DS	F	Object handle
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## Appel Visual Basic

Windows

```
MQOPEN Hconn, ObjDesc, Options, Hobj, CompCode, Reason
```

Déclarez les paramètres comme suit :

```
Dim Hconn As Long 'Connection handle'
Dim ObjDesc As MQOD 'Object descriptor'
Dim Options As Long 'Options that control the action of MQOPEN'
Dim Hobj As Long 'Object handle'
Dim CompCode As Long 'Completion code'
Dim Reason As Long 'Reason code qualifying CompCode'
```

## MQPUT-Insertion d'un message

L'appel MQPUT place un message dans une file d'attente ou une liste de distribution, ou dans une rubrique. La file d'attente, la liste de distribution ou la rubrique doit déjà être ouverte.

### Syntaxe

MQPUT (*Hconn, Hobj, MsgDesc, PutMsgOpts, BufferLength, Buffer, CompCode, Raison*)

### Paramètres

#### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de Hconn a été renvoyée par un appel MQCONN ou MQCONNX précédent.

**z/OS** Sur les applications z/OS for CICS, l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour Hconn:

#### MQHC\_DEF\_HCONN

Descripteur de connexion par défaut.

#### HOBJ

Type : MQHOBJ - entrée

Cet identificateur représente la file d'attente à laquelle le message est ajouté ou la rubrique dans laquelle le message est publié. La valeur de Hobj a été renvoyée par un appel MQOPEN précédent qui a spécifié l'option MQOO\_OUTPUT.

## MsgDesc

Type : MQMD - entrée/sortie

Cette structure décrit les attributs du message envoyé et reçoit des informations sur le message une fois la demande d'insertion terminée. Voir «[MQMD-Descripteur de message](#)», à la page 438 pour des détails.

Si l'application fournit un MQMD version-1, les données de message peuvent être préfixées avec une structure MQMDE afin de spécifier des valeurs pour les zones qui existent dans le MQMD version-2 mais pas dans version-1. La zone *Format* du MQMD doit être définie sur MQFMT\_MD\_EXTENSION pour indiquer qu'un MQMDE est présent. Pour plus d'informations, voir «[MQMDE-Extension de descripteur de message](#)», à la page 492.

L'application n'a pas besoin de fournir une structure MQMD si un descripteur de message valide est fourni dans les zones *OriginalMsgHandle* ou *NewMsgHandle* de la structure MQPMO. Si aucune donnée n'est fournie dans l'un de ces champs, le descripteur du message est extrait du descripteur associé aux descripteurs de message.

Si vous utilisez ou prévoyez d'utiliser des exits API, nous vous recommandons de fournir explicitement une structure MQMD et de ne pas utiliser les descripteurs de message associés aux descripteurs de message. En effet, l'exit API associé à l'appel MQPUT ou MQPUT1 ne parvient pas à déterminer les valeurs MQMD utilisées par le gestionnaire de files d'attente pour exécuter la demande MQPUT ou MQPUT1.

## PutMsgOpts

Type: MQPMO-entrée/sortie

Pour plus d'informations, voir «[MQPMO-Options de message d'insertion](#)», à la page 522.

## BufferLength

Type : MQLONG - entrée

Longueur du message dans *Buffer*. La valeur zéro est valide et indique que le message ne contient aucune donnée d'application. La limite supérieure de *BufferLength* dépend de différents facteurs:

- Si la destination est une file d'attente locale ou est convertie en file d'attente locale, la limite supérieure varie selon que:
  - Le gestionnaire de files d'attente local prend en charge la segmentation.
  - L'application émettrice spécifie l'indicateur qui permet au gestionnaire de files d'attente de segmenter le message. Cet indicateur est MQMF\_SEGMENTATION\_ALLOWED et peut être spécifié dans un MQMD version-2 ou dans un MQMDE utilisé avec un MQMD version-1.

Si ces deux conditions sont satisfaites, *BufferLength* ne peut pas dépasser 999 999 999 moins la valeur de la zone *Offset* dans MQMD. Le message logique le plus long pouvant être inséré est donc 999 999 999 octets (lorsque *Offset* est égal à zéro). Toutefois, les contraintes de ressources imposées par le système d'exploitation ou l'environnement dans lequel l'application s'exécute peuvent entraîner une limite inférieure.

Si l'une des conditions précédentes ou les deux conditions précédentes ne sont pas satisfaites, *BufferLength* ne peut pas dépasser la plus petite des valeurs de l'attribut **MaxMsgLength** de la file d'attente et de l'attribut **MaxMsgLength** du gestionnaire de files d'attente.

- Si la destination est une file d'attente éloignée ou est convertie en file d'attente éloignée, les conditions applicables aux files d'attente locales s'appliquent, mais au niveau de chaque gestionnaire de files d'attente par lequel le message doit passer pour atteindre la file d'attente de destination ; en particulier:
  1. File d'attente de transmission locale utilisée pour stocker temporairement le message sur le gestionnaire de files d'attente local
  2. Files d'attente de transmission intermédiaires (le cas échéant) utilisées pour stocker les messages au niveau des gestionnaires de files d'attente sur la route entre les gestionnaires de files d'attente local et de destination
  3. File d'attente de destination sur le gestionnaire de files d'attente de destination

Le message le plus long pouvant être inséré est donc régi par les files d'attente et les gestionnaires de files d'attente les plus restrictifs.

Lorsqu'un message se trouve dans une file d'attente de transmission, des informations supplémentaires se trouvent avec les données de message, ce qui réduit la quantité de données d'application pouvant être transférées. Dans cette situation, soustrayez les octets MQ\_MSG\_HEADER\_LENGTH des valeurs MaxMsgLength des files d'attente de transmission lors de la détermination de la limite pour BufferLength.

**Remarque :** Seul le non-respect de la condition 1 peut être diagnostiqué de manière synchrone (avec le code anomalie MQRC\_MSG\_TOO\_BIG\_FOR\_Q ou MQRC\_MSG\_TOO\_BIG\_FOR\_Q\_MGR) lorsque le message est inséré. Si les conditions 2 ou 3 ne sont pas satisfaites, le message est redirigé vers une file d'attente de rebut (message non distribué), soit au niveau d'un gestionnaire de files d'attente intermédiaire, soit au niveau du gestionnaire de files d'attente de destination. Si cela se produit, un message de rapport est généré si l'expéditeur en a demandé un.

### Mémoire tampon

Type: MQBYTEExBufferLength-input

Il s'agit d'une mémoire tampon contenant les données d'application à envoyer. La mémoire tampon doit être alignée sur une limite appropriée à la nature des données du message. L'alignement sur 4 octets convient à la plupart des messages (y compris les messages contenant des structures d'en-tête IBM MQ), mais certains messages peuvent nécessiter un alignement plus strict. Par exemple, un message contenant un entier binaire de 64 bits peut nécessiter un alignement de 8 octets.

Si Buffer contient des données de type caractère ou numérique, définissez les zones CodedCharSetId et Encoding du paramètre **MsgDesc** sur les valeurs appropriées aux données ; cela permet au destinataire du message de convertir les données (si nécessaire) en jeu de caractères et en codage utilisés par le destinataire.

**Remarque :** Tous les autres paramètres de l'appel MQPUT doivent être dans le jeu de caractères et le codage du gestionnaire de files d'attente local (indiqués par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et MQENC\_NATIVE).

Dans le langage de programmation C, le paramètre est déclaré comme un pointeur vers vide ; l'adresse de tout type de données peut être spécifiée comme paramètre.

Si la valeur du paramètre **BufferLength** est zéro, il n'est pas fait référence à Buffer ; dans ce cas, l'adresse de paramètre transmise par les programmes écrits en assembleur C ou System/390 peut être null.

### CompCode

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_WARNING**

Avertissement (achèvement partiel).

#### **MQCC\_FAILED**

Echec de l'appel.

### Motif

Type : MQLONG - sortie

Code anomalie qualifiant CompCode.

Si CompCode est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si CompCode est MQCC\_WARNING:

**MQRC\_INCOMPLETE\_GROUP**

(2241, X'8C1') Groupe de messages non complet.

**MQRC\_INCOMPLETE\_MSG**

(2242, X'8C2') Message logique non complet.

**MQRC\_INCONSISTENT\_PERSISTENCE**

(2185, X'889') Spécification de persistance incohérente.

**MQRC\_INCONSISTENT\_UOW**

(2245, X'8C5') Spécification d'unité d'oeuvre incohérente.

**MQRC\_MULTIPLE\_MOTIFS**

(2136, X'858') Plusieurs codes anomalie ont été renvoyés.

**MQRC\_PRIORITY\_MAXIMAL\_DÉPASSEMENTS**

(2049, X'801') La priorité de message dépasse la valeur maximale prise en charge.

**MQRC\_UNKNOWN\_REPORT\_OPTION**

(2104, X'838') Option (s) de rapport dans le descripteur de message non reconnue.

Si CompCode est MQCC\_FAILED:

**MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') Adaptateur non disponible.

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

**MQRC\_ALIAS\_TARGTYPE\_CHANGED**

(2480, X'09B0') Le type de cible d'abonnement a été modifié de file d'attente en objet de rubrique.

**MQRC\_API\_EXIT\_ERROR**

(2374, X'946') Echec de l'exit API.

**MQRC\_API\_EXIT\_LOAD\_ERROR**

(2183, X'887') Impossible de charger l'exit API.

**MQRC\_ASID\_MISMATCH**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

**MQRC\_BACKED\_OUT**

(2003, X'7D3') Unité d'oeuvre annulée.

**MQRC\_BUFFER\_ERROR**

(2004, X'7D4') Paramètre de tampon non valide.

**MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

**MQRC\_CALL\_INTERROMPUE**

(2549, X'9F5') MQPUT ou MQCMIT a été interrompu et le traitement de la reconnexion ne peut pas rétablir un résultat défini.

**MQRC\_CF\_NOT\_XX\_ENCODE\_CASE\_ONE disponible**

(2345, X' 929') Unité de couplage non disponible.

**MQRC\_CF\_STRUC\_FAILED**

(2373, X'945') Echec de la structure de l'unité de couplage.

**MQRC\_CF\_STRUC\_IN\_USE**

(2346, X'92A') Structure de l'unité de couplage en cours d'utilisation.

**MQRC\_CFGR\_ERREUR**

(2416, X' 970') La structure de paramètre de groupe PCF MQCFGR dans les données de message n'est pas valide.

**MQRC\_CFH\_ERREUR**

(2235, X'8BB') Structure d'en-tête PCF non valide.

**MQRC\_CFIF\_ERREUR**

(2414, X'96E') La structure du paramètre de filtre d'entier PCF dans les données de message n'est pas valide.

**MQRC\_CFIL\_ERREUR**

(2236, X'8BC') Structure de paramètre de liste d'entiers PCF ou PCIF\*64 structure de paramètre de liste d'entiers non valide.

**MQRC\_CFIN\_ERREUR**

(2237, X'8BD') Structure de paramètre d'entier PCF ou PCIF\*64 structure de paramètre d'entier non valide.

**MQRC\_CFSF\_ERREUR**

(2415, X'96F') La structure de paramètre de filtre de chaîne PCF dans les données de message n'est pas valide.

**MQRC\_CFSL\_ERREUR**

(2238, X'8BE') Structure de paramètre de liste de chaînes PCF incorrecte.

**MQRC\_CFST\_ERREUR**

(2239, X'8BF') Structure de paramètre de chaîne PCF incorrecte.

**MQRC\_CICS\_WAIT\_FAILED**

(2140, X'85C') Demande d'attente rejetée par CICS.

**MQRC\_CLUSTER\_EXIT\_ERREUR**

(2266, X'8DA') L'exit de charge de travail de cluster a échoué.

**Erreur de résolution MQRC\_CLUSTER\_RESOLUTION\_ERROR**

(2189, X'88D') La résolution du nom de cluster a échoué.

**MQRC\_CLUSTER\_RESOURCE\_ERROR**

(2269, X'8DD') Erreur de ressource de cluster.

**MQRC\_COD\_NOT\_VALID\_FOR\_XCF\_Q**

(2106, X'83A') L'option de rapport COD n'est pas valide pour la file d'attente XCF.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**MQRC\_CONNECTION\_NOT\_AUTHORIZED**

(2217, X'8A9') Non autorisé pour la connexion.

**MQRC\_CONNECTION QUIESCING**

(2202, X'89A') Connexion en cours de mise au repos.

**MQRC\_CONNECTION\_STOPPING**

(2203, X'89B') Connexion en cours d'arrêt.

**MQRC\_CONTENT\_ERREUR**

2554 (X'09FA') Le contenu du message n'a pas pu être analysé pour déterminer si le message doit être distribué à un abonné avec un sélecteur de message étendu.

**MQRC\_CONTEXT\_HANDLE\_ERREUR**

(2097, X'831') L'identificateur de file d'attente auquel il est fait référence ne sauvegarde pas le contexte.

**MQRC\_CONTEXT\_NOT\_AVAILABLE**

(2098, X'832') Contexte non disponible pour le descripteur de file d'attente référencé.

**MQRC\_DATA\_LENGTH\_ERROR**

(2010, X'7DA') Paramètre de longueur des données non valide.

**MQRC\_DH\_ERREUR**

(2135, X'857') Structure d'en-tête de distribution incorrecte.

**MQRC\_DLH\_ERREUR**

(2141, X'85D') Structure d'en-tête de rebut non valide.

**MQRC\_EPH\_ERREUR**

(2420, X' 974') Structure PCF intégrée incorrecte.

**MQRC\_EXPIRY\_ERROR**

(2013, X'7DD') Heure d'expiration non valide.

**MQRC\_ERREUR\_FEEDBACK\_ERROR**

(2014, X'7DE') Code retour non valide.

**MQRC\_GLOBAL\_UOW\_CONFLICT**

(2351, X'92F') Conflit entre les unités d'oeuvre globales.

**MQRC\_GROUP\_ID\_ERREUR**

(2258, X'8D2') Identificateur de groupe non valide.

**MQRC\_HANDLE\_IN\_USE\_FOR\_UOW**

(2353, X'931') Descripteur en cours d'utilisation pour l'unité d'oeuvre globale.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') Descripteur de connexion non valide.

**MQRC\_HEADER\_ERREUR**

(2142, X'85E') Structure d'en-tête MQ non valide.

**MQRC\_HOBJ\_ERROR**

(2019, X'7E3') Descripteur d'objet non valide.

**MQRC\_IIH\_ERREUR**

(2148, X'864') La structure d'en-tête d'information IMS n'est pas valide.

**MQRC\_INCOMPLETE\_GROUP**

(2241, X'8C1') Groupe de messages non complet.

**MQRC\_INCOMPLETE\_MSG**

(2242, X'8C2') Message logique non complet.

**MQRC\_INCONSISTENT\_PERSISTENCE**

(2185, X'889') Spécification de persistance incohérente.

**MQRC\_INCONSISTENT\_UOW**

(2245, X'8C5') Spécification d'unité d'oeuvre incohérente.

**MQRC\_LOCAL\_UOW\_CONFLICT**

(2352, X'930') Conflit entre l'unité d'oeuvre globale et l'unité d'oeuvre locale.

**MQRC\_MD\_ERROR**

(2026, X'7EA') Descripteur de message non valide.

**MQRC\_MDE\_ERREUR**

(2248, X'8C8') Extension de descripteur de message non valide.

**MQRC\_MISSING\_REPLY\_TO\_Q**

(2027, X'7EB') Une file d'attente de réponse manquante ou MQPMO\_SUPPRESS\_REPLYTO a été utilisée

**MQRC\_WIH**

(2332, X'91C') Les données de message ne commencent pas par MQWIH.

**MQRC\_MSG\_FLAGS\_ERREUR**

(2249, X'8C9') Indicateurs de message non valides.

**MQRC\_MSG\_SEQ\_NUMBER\_ERROR**

(2250, X'8CA') Numéro de séquence de message non valide.

**MQRC\_MSG\_TOO\_BIG\_FOR\_Q**

(2030, X'7EE') Longueur de message supérieure au maximum pour la file d'attente.

**MQRC\_MSG\_TOO\_BIG\_FOR\_Q\_MGR**

(2031, X'7EF') Longueur de message supérieure à la longueur maximale pour le gestionnaire de files d'attente.

**MQRC\_MSG\_TYPE\_ERREUR**

(2029, X'7ED') Le type de message dans le descripteur de message n'est pas valide.

**MQRC\_MULTIPLE\_MOTIFS**

(2136, X'858') Plusieurs codes anomalie ont été renvoyés.

**MQRC\_NO\_DESTINATIONS\_AVAILABLE**

(2270, X'8DE') Aucune file d'attente de destination disponible.

**MQRC\_NOT\_OPEN\_FOR\_OUTPUT**

(2039, X'7F7') File d'attente non ouverte pour la sortie.

**MQRC\_NOT\_OPEN\_FOR\_PASS\_ALL**

(2093, X'82D') File d'attente non ouverte pour le passage de tous les contextes.

**MQRC\_NOT\_OPEN\_FOR\_PASS\_IDENT**

(2094, X'82E') File d'attente non ouverte pour le contexte d'identité de transmission.

**MQRC\_NOT\_OPEN\_FOR\_SET\_ALL**

(2095, X'82F') File d'attente non ouverte pour tous les contextes.

**MQRC\_NOT\_OPEN\_FOR\_SET\_IDENT**

(2096, X'830') File d'attente non ouverte pour le contexte d'identité défini.

**MQRC\_OBJECT\_CHANGED**

(2041, X'7F9') Définition d'objet modifiée depuis son ouverture.

**MQRC\_OBJECT\_DAMAGED**

(2101, X'835') Objet endommagé.

**ERREUR MQRC\_OFFSET\_ERROR**

(2251, X'8CB') Position de segment de message non valide.

**MQRC\_OPEN\_FAILED**

(2137, X'859') L'ouverture de l'objet a échoué.

**MQRC\_OPTIONS\_ERROR**

(2046, X'7FE') Options incorrectes ou incohérentes.

**MQRC\_ORIGINAL\_LENGTH\_ERROR**

(2252, X'8CC') Longueur d'origine incorrecte.

**MQRC\_PAGESET\_ERROR**

(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.

**MQRC\_PAGESET\_FULL**

(2192, X'890') Support de stockage externe saturé.

**MQRC\_PCF\_ERREUR**

(2149, X'865') Structures PCF incorrectes.

**MQRC\_XX\_ENCODE\_CASE\_ONE err\_persistent**

(2047, X'7FF') Persistance non valide.

**MQRC\_PERSISTENT\_NOT\_ALLOWED**

(2048, X'800') La file d'attente ne prend pas en charge les messages persistants.

**MQRC\_PMO\_ERREUR**

(2173, X'87D') La structure des options d'insertion de message n'est pas valide.

**MQRC\_PMO\_RECORD\_FLAGS\_ERROR**

(2158, X'86E') Les indicateurs d'enregistrement de message d'insertion ne sont pas valides.

**MQRC\_PRIORITY\_ERROR**

(2050, X'802') Priorité de message non valide.

**MQRC\_PUBLICATION\_FAILURE**

(2502, X'9C6') La publication n'a été distribuée à aucun des abonnés.

**MQRC\_PUT\_INHIBÉ**

(2051, X'803') Les appels d'insertion sont interdits pour la file d'attente, pour la file d'attente dans laquelle cette file d'attente est résolue ou pour la rubrique.

**MQRC\_PUT\_MSG\_RECORDS\_ERROR**

(2159, X'86F') Enregistrements de message d'insertion non valides.

**MQRC\_PUT\_NOT\_RETENU**

(2479, X'09AF') La publication n'a pas pu être conservée

**MQRC\_Q\_DELETED**

(2052, X'804') File d'attente supprimée.

**MQRC\_Q\_FULL**

(2053, X'805') La file d'attente contient déjà le nombre maximal de messages.

**MQRC\_Q\_MGR\_NAME\_ERROR**

(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**

(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

**MQRC\_Q\_MGR QUIESCING**

(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**MQRC\_Q\_ESPACE\_NON\_DISPONIBLE**

(2056, X'808') Aucun espace disponible sur le disque pour la file d'attente.

**Echec de MQRC\_RECONNECT\_FAILED**

(2548, X'9F4') Après la reconnexion, une erreur s'est produite lors de la réinstallation des descripteurs d'une connexion reconnectable.

**MQRC\_RECS\_Présentateur-ERREUR**

(2154, X'86A') Nombre d'enregistrements présents non valide.

**MQRC\_REPORT\_OPTIONS\_ERREUR**

(2061, X'80D') Les options de rapport dans le descripteur de message ne sont pas valides.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') Ressources système disponibles insuffisantes.

**MQRC\_RESPONSE\_RECORDS\_ERROR**

(2156, X'86C') Enregistrements de réponse non valides.

**MQRC\_RFH\_ERREUR**

(2334, X'91E') Structure MQRFH ou MQRFH2 non valide.

**MQRC\_RMH\_ERREUR**

(2220, X'8AC') La structure d'en-tête de message de référence n'est pas valide.

**MQRC\_SEGMENT\_LONGUEUR\_ZÉRO**

(2253, X'8CD') La longueur des données dans le segment de message est égale à zéro.

**MQRC\_SEGMENTS\_NON\_PRIS en charge**

(2365, X'93D') Segments non pris en charge.

**MQRC\_SELECTION\_NOT\_AVAILABLE**

2551 (X'09F7') Il existe un abonné possible pour la publication, mais le gestionnaire de files d'attente ne peut pas vérifier si la publication doit être envoyée à l'abonné.

**MQRC\_STOPPED\_BY\_CLUSTER\_EXIT**

(2188, X'88C') Appel rejeté par l'exit de charge de travail du cluster.

**MQRC\_STORAGE\_CLASS\_ERROR**

(2105, X'839') Erreur de classe de stockage.

**MQRC\_STORAGE\_MEDIUM\_FULL**

(2192, X'890') Support de stockage externe saturé.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') Mémoire disponible insuffisante.

**MQRC\_SUPPRESSED\_BY\_EXIT**

(2109, X'83D') Appel supprimé par le programme d'exit.

**MQRC\_SYNCPOINT\_LIMIT\_REACHED**

(2024, X'7E8') Aucun autre message pouvant être traité au sein de l'unité d'oeuvre en cours.

**MQRC\_SYNCPOINT\_NOT\_AVAILABLE**

(2072, X'818') Prise en charge du point de synchronisation non disponible.



**MQRC\_TM\_ERREUR**

(2265, X'8D9') La structure du message de déclenchement n'est pas valide.

**MQRC\_TMC\_ERREUR**

(2191, X'88F') La structure de message du déclencheur de caractères n'est pas valide.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') Une erreur inattendue s'est produite.

**MQRC\_UOW\_ENLISTMENT\_ERROR**

(2354, X'932') Echec de l'inscription dans l'unité d'oeuvre globale.

**MQRC\_UOW\_MIX\_NOT\_SUPPORTED**

(2355, X'933') Mélange d'appels d'unité d'oeuvre non pris en charge.

**MQRC\_UOW\_NOT\_AVAILABLE**

(2255, X'8CF') Unité d'oeuvre non disponible pour le gestionnaire de files d'attente à utiliser.

**MQRC\_WIH\_ERREUR**

(2333, X'91D') Structure MQWIH non valide.

**MQRC\_WRONG\_MD\_VERSION**

(2257, X'8D1') Version MQMD fournie non valide.

**MQRC\_XQH\_ERREUR**

(2260, X'8D4') Structure d'en-tête de file d'attente de transmission non valide.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Remarques sur l'utilisation des rubriques

1. Les remarques suivantes s'appliquent à l'utilisation des rubriques:

a. Lors de l'utilisation de MQPUT pour publier des messages sur une rubrique, où un ou plusieurs abonnés à cette rubrique ne peuvent pas recevoir la publication en raison d'un problème avec leur file d'attente d'abonné (par exemple, elle est saturée), le code anomalie renvoyé à l'appel MQPUT et le comportement de distribution dépendent de la définition des attributs PMSGDLV ou NPMSGDLV sur le TOPIC. La distribution d'une publication dans la file d'attente des messages non livrés lorsque MQRO\_DEAD\_LETTER\_Q est spécifié, ou la suppression du message lorsque MQRO\_DISCARD\_MSG est spécifié, est considérée comme une distribution réussie du message. Si aucune des publications n'est distribuée, MQPUT est renvoyé avec MQRC\_PUBLICATION\_FAILURE. Cela peut se produire dans les cas suivants:

- Un message est publié sur un TOPIC avec PMSGDLV ou NPMSGDLV (en fonction de la persistance du message) défini sur ALL et tout abonnement (durable ou non) possède une file d'attente qui ne peut pas recevoir la publication.
- Un message est publié sur un TOPIC avec PMSGDLV ou NPMSGDLV (en fonction de la persistance du message) défini sur ALLDUR et un abonnement durable possède une file d'attente qui ne peut pas recevoir la publication.

MQPUT peut être renvoyé avec MQRC\_NONE même si les publications n'ont pas pu être distribuées à certains abonnés dans les cas suivants:

- Un message est publié sur un TOPIC avec PMSGDLV ou NPMSGDLV (en fonction de la persistance du message) défini sur ALLAVAIL et tout abonnement, durable ou non, possède une file d'attente qui ne peut pas recevoir la publication.
- Un message est publié sur un TOPIC avec PMSGDLV ou NPMSGDLV (en fonction de la persistance du message) défini sur ALLDUR et un abonnement non durable possède une file d'attente qui ne peut pas recevoir la publication.

Vous pouvez utiliser l'attribut de rubrique USEDQLQ pour déterminer si la file d'attente de rebut est utilisée lorsque les messages de publication ne peuvent pas être distribués à leur file d'attente de souscription correcte. Pour plus d'informations sur l'utilisation de USEDQLQ, voir [DEFINE TOPIC](#).

b. S'il n'y a pas d'abonnés à la rubrique utilisée, le message publié n'est envoyé à aucune file d'attente et est supprimé. Peu importe que le message soit persistant ou non persistant, qu'il ait une

expiration illimitée ou qu'il ait une heure d'expiration, il est tout de même supprimé s'il n'y a pas d'abonnés. La seule exception à cette règle est si le message doit être conservé, auquel cas, bien qu'il ne soit pas envoyé aux files d'attente des abonnés, il est stocké par rapport à la rubrique à distribuer aux nouveaux abonnements ou aux abonnés qui demandent des publications conservées à l'aide de MQSUBRQ.

## MQPUT et MQPUT1

Vous pouvez utiliser les appels MQPUT et MQPUT1 pour placer des messages dans une file d'attente ; l'appel à utiliser dépend des circonstances

- Utilisez l'appel MQPUT pour placer plusieurs messages dans la même file d'attente.

Un appel MQOPEN spécifiant l'option MQOO\_OUTPUT est émis en premier, suivi d'une ou de plusieurs demandes MQPUT pour ajouter des messages à la file d'attente ; enfin, la file d'attente est fermée avec un appel MQCLOSE. Cela offre de meilleures performances que l'utilisation répétée de l'appel MQPUT1 .

- Utilisez l'appel MQPUT1 pour placer un seul message dans une file d'attente.

Cet appel encapsule les appels MQOPEN, MQPUT et MQCLOSE en un seul appel, ce qui réduit le nombre d'appels qui doivent être émis.

## Files d'attente de destination

Les remarques suivantes s'appliquent à l'utilisation des files d'attente de destination:

1. Si une application place une séquence de messages dans la même file d'attente sans utiliser de groupes de messages, l'ordre de ces messages est conservé si les conditions détaillées sont remplies. Certaines conditions s'appliquent aux files d'attente de destination locales et éloignées ; d'autres conditions s'appliquent uniquement aux files d'attente de destination éloignées.


### Conditions qui s'appliquent aux files d'attente de destination locales et distantes

- Tous les appels MQPUT se trouvent dans la même unité de travail, ou aucun d'entre eux ne se trouve dans une unité de travail.

Sachez que lorsque des messages sont placés dans une file d'attente particulière au sein d'une seule unité d'oeuvre, les messages provenant d'autres applications peuvent être associés à la séquence de messages de la file d'attente.

- Tous les appels MQPUT sont effectués à l'aide du même descripteur d'objet *Hobj*.

Dans certains environnements, la séquence de messages est également conservée lorsque des descripteurs d'objet différents sont utilisés, si les appels sont effectués à partir de la même application. La signification de *même application* est déterminée par l'environnement:

–  Sous z/OS, l'application est:

- Pour CICS, la tâche CICS
- Pour IMS, la tâche
- Pour le lot z/OS , la tâche

–  Sous IBM i, l'application est le travail.

–  Sous AIX, Linux, and Windows, l'application est l'unité d'exécution.

- Les messages ont tous la même priorité.
- Les messages ne sont pas insérés dans une file d'attente de cluster avec MQOO\_BIND\_NOT\_FIXED spécifié (ou avec MQOO\_BIND\_AS\_Q\_DEF en vigueur lorsque l'attribut de file d'attente DefBind a la valeur MQBND\_BIND\_NOT\_FIXED).

### Conditions supplémentaires qui s'appliquent aux files d'attente de destination éloignées

- Il n'existe qu'un seul chemin entre le gestionnaire de files d'attente émetteur et le gestionnaire de files d'attente cible.

Si certains messages de la séquence peuvent se trouver dans un chemin différent (par exemple, en raison de la reconfiguration, de l'équilibrage du trafic ou de la sélection du chemin en fonction de la taille des messages), l'ordre des messages sur le gestionnaire de files d'attente de destination ne peut pas être garanti.

- Les messages ne sont pas placés temporairement dans des files d'attente de rebut au niveau des gestionnaires de files d'attente d'envoi, intermédiaire ou de destination.

Si un ou plusieurs messages sont placés temporairement dans une file d'attente de rebut (par exemple, parce qu'une file d'attente de transmission ou la file d'attente de destination est temporairement saturée), les messages peuvent arriver dans la file d'attente de destination hors séquence.

- Les messages sont tous persistants ou non persistants.

Si l'attribut **NonPersistentMsgSpeed** d'un canal sur la route entre les gestionnaires de files d'attente d'envoi et de destination est défini sur MQNPMS\_FAST, les messages non persistants peuvent prendre le pas sur les messages persistants, ce qui a pour conséquence que l'ordre des messages persistants par rapport aux messages non persistants n'est pas conservé. Cependant, l'ordre des messages persistants les uns par rapport aux autres et des messages non persistants les uns par rapport aux autres est conservé.

Si ces conditions ne sont pas remplies, vous pouvez utiliser des groupes de messages pour conserver l'ordre des messages, mais cela nécessite que les applications d'envoi et de réception utilisent la prise en charge du regroupement de messages. Pour plus d'informations sur les groupes de messages, voir :

- [MQMD - Zone MsgFlags](#)
- [MQPMO\\_LOGICAL\\_ORDER](#)
- [MQGMO\\_LOGICAL\\_ORDER](#)

## Listes de Distribution

Les remarques suivantes s'appliquent à l'utilisation des listes de distribution.

Les listes de distribution sont prises en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour IBM MQ MQI clients connecté à ces systèmes.

1. Vous pouvez placer des messages dans une liste de distribution à l'aide d'une instance version-1 ou d'une instance version-2 MQPMO. Si vous utilisez un MQPMO version-1 (ou un MQPMO version-2 avec RecsPresent égal à zéro), l'application ne peut pas fournir d'enregistrements de message d'insertion ni d'enregistrements de réponse. Vous ne pouvez pas identifier les files d'attente qui rencontrent des erreurs si le message est envoyé avec succès à certaines files d'attente de la liste de distribution et non à d'autres.

Si l'application fournit des enregistrements de message d'insertion ou des enregistrements de réponse, définissez la zone Version sur MQPMO\_VERSION\_2.

Vous pouvez également utiliser un MQPMO version-2 pour envoyer des messages à une file d'attente unique qui ne figure pas dans une liste de distribution, en vous assurant que RecsPresent est égal à zéro.

2. Les paramètres de code achèvement et de code anomalie sont définis comme suit:

- Si les insertions dans les files d'attente de la liste de distribution aboutissent ou échouent de la même manière, les paramètres de code achèvement et de code raison sont définis pour décrire le

résultat commun. Les enregistrements de réponse MQRR (s'ils sont fournis par l'application) ne sont pas définis dans ce cas.

Par exemple, si chaque insertion aboutit, le code achèvement et le code anomalie sont définis sur MQCC\_OK et MQRC\_NON; si chaque insertion échoue car toutes les files d'attente sont interdites pour les insertions, les paramètres sont définis sur MQCC\_FAILED et MQRC\_PUT\_INHIBÉ.

- Si les insertions dans les files d'attente de la liste de distribution n'aboutissent pas toutes ou échouent de la même manière:
  - Le paramètre de code achèvement est défini sur MQCC\_WARNING si au moins une insertion a abouti et sur MQCC\_FAILED si toutes les opérations ont échoué.
  - Le paramètre de code anomalie est défini sur MQRC\_MULTIPLE\_MOTIFS.
  - Les enregistrements de réponse (s'ils sont fournis par l'application) sont définis sur les codes achèvement individuels et les codes raison des files d'attente de la liste de distribution.

Si l'insertion dans une destination échoue en raison de l'échec de l'ouverture de cette destination, les zones de l'enregistrement de réponse sont définies sur MQCC\_FAILED et MQRC\_OPEN\_FAILED; cette destination est incluse dans InvalidDestCount.

3. Si une destination de la liste de distribution se résout en une file d'attente locale, le message est placé dans cette file d'attente sous la forme normale (c'est-à-dire qu'il ne s'agit pas d'un message de liste de distribution). Si plusieurs destinations sont résolues dans la même file d'attente locale, un message est placé dans la file d'attente pour chaque destination.

Si une destination de la liste de distribution est résolue en file d'attente éloignée, un message est placé dans la file d'attente de transmission appropriée. Lorsque plusieurs destinations sont résolues dans la même file d'attente de transmission, un seul message de liste de distribution contenant ces destinations peut être placé dans la file d'attente de transmission, même si ces destinations n'étaient pas adjacentes dans la liste de destinations fournie par l'application. Toutefois, cette opération ne peut être effectuée que si la file d'attente de transmission prend en charge les messages de liste de distribution (voir [DistLists](#)).

Si la file d'attente de transmission ne prend pas en charge les listes de distribution, une copie du message au format normal est placée dans la file d'attente de transmission pour chaque destination qui utilise cette file d'attente de transmission.

Si une liste de distribution avec les données de message d'application est trop grande pour une file d'attente de transmission, le message de liste de distribution est divisé en messages de liste de distribution plus petits, chacun contenant moins de destinations. Si les données de message d'application ne correspondent qu'à la file d'attente, les messages de liste de distribution ne peuvent pas être utilisés du tout et le gestionnaire de files d'attente génère une copie du message au format normal pour chaque destination qui utilise cette file d'attente de transmission.

Si différentes destinations ont une priorité de message ou une persistance de message différente (cela peut se produire lorsque l'application spécifie MQPRI\_PRIORITY\_AS\_Q\_DEF ou MQPER\_PERSISTENCE\_AS\_Q\_DEF), les messages ne sont pas conservés dans le même message de liste de distribution. Au lieu de cela, le gestionnaire de files d'attente génère autant de messages de liste de distribution que nécessaire pour prendre en charge les valeurs de priorité et de persistance différentes.

4. Une insertion dans une liste de distribution peut entraîner:

- un message de liste de distribution unique, ou
- Un certain nombre de messages de liste de distribution plus petits, ou
- Une combinaison de messages de liste de distribution et de messages normaux, ou
- Messages normaux uniquement.

Ce qui précède se produit selon que:

- Les destinations de la liste sont locales, distantes ou un mélange.
- Les destinations ont la même priorité de message et la même persistance de message.

- Les files d'attente de transmission peuvent contenir des messages de liste de distribution.
- La longueur maximale des messages des files d'attente de transmission est suffisante pour accueillir le message sous forme de liste de distribution.

Toutefois, quel que soit le message ci-dessus, chaque message *physique* résultant (c'est-à-dire chaque message normal ou de liste de distribution résultant de l'insertion) compte comme *un seul* message lorsque:

- Vérifier si l'application a dépassé le nombre maximal de messages autorisé dans une unité de travail (voir l'attribut du gestionnaire de files d'attente **MaxUncommittedMsgs** ).
  - Vérification du respect des conditions de déclenchement.
  - Incrémentation du nombre de lignes de la file d'attente et vérification du dépassement du nombre maximal de lignes de la file d'attente.
5. Toute modification apportée aux définitions de file d'attente qui aurait entraîné la non-validité d'un descripteur si les files d'attente avaient été ouvertes individuellement (par exemple, une modification du chemin de résolution) n'entraîne pas la non-validité du descripteur de liste de distribution. Toutefois, elle entraîne un échec pour cette file d'attente particulière lorsque le descripteur de liste de distribution est utilisé lors d'un appel MQPUT ultérieur.

## En-têtes

Si un message est inséré avec une ou plusieurs structures d'en-tête IBM MQ au début des données de message d'application, le gestionnaire de files d'attente effectue certaines vérifications sur les structures d'en-tête pour vérifier qu'elles sont valides. Si le gestionnaire de files d'attente détecte une erreur, l'appel échoue avec un code anomalie approprié. Les contrôles effectués varient en fonction des structures particulières présentes:

- Les vérifications ne sont effectuées que si une version-2 ou une version ultérieure de MQMD est utilisée sur l'appel MQPUT ou MQPUT1 . Les vérifications ne sont pas effectuées si un MQMD version-1 est utilisé, même si un MQMDE est présent au début des données du message.
- Les structures qui ne sont pas prises en charge par le gestionnaire de files d'attente local et les structures qui suivent le premier MQDLH dans le message ne sont pas validées.
- Les structures MQDH et MQMDE sont entièrement validées par le gestionnaire de files d'attente.
- Les autres structures sont partiellement validées par le gestionnaire de files d'attente (toutes les zones ne sont pas vérifiées).

Les vérifications générales effectuées par le gestionnaire de files d'attente sont les suivantes:

- La zone `StrucId` doit être valide.
- La zone `Version` doit être valide.
- La zone `StrucLength` doit spécifier une valeur suffisamment grande pour inclure la structure plus les données de longueur variable qui font partie de la structure.
- La zone `CodedCharSetId` ne doit pas avoir la valeur zéro ou une valeur négative non valide (MQCCSI\_DEFAULT, MQCCSI\_EMBEDDED, MQCCSI\_Q\_MGR et MQCCSI\_UNDEFINED ne sont pas valides dans la plupart des structures d'en-tête IBM MQ ).
- Le paramètre **BufferLength** de l'appel doit spécifier une valeur suffisamment élevée pour inclure la structure (la structure ne doit pas s'étendre au-delà de la fin du message).

Outre les contrôles généraux des structures, les conditions suivantes doivent être remplies:

- La somme des longueurs des structures d'un message PCF doit être égale à la longueur spécifiée par le paramètre **BufferLength** dans l'appel MQPUT ou MQPUT1 . Un message PCF est un message dont le nom de format est MQFMT\_ADMIN, MQFMT\_EVENT ou MQFMT\_PCF.
- Une structure IBM MQ ne doit pas être tronquée, sauf dans les cas suivants où les structures tronquées sont autorisées:
  - Messages qui sont des messages de rapport.

- Messages PCF.
- Messages contenant une structure MQDLH. (Les structures qui suivent le premier MQDLH peuvent être tronquées ; les structures qui précèdent le MQDLH ne le peuvent pas.)
- Une structure IBM MQ ne doit pas être divisée sur deux ou plusieurs segments ; elle doit être entièrement contenue dans un segment.

## Mémoire tampon

Pour le langage de programmation Visual Basic, les points suivants s'appliquent :

- Si la taille du paramètre **Buffer** est inférieure à la longueur spécifiée par le paramètre **BufferLength**, l'appel échoue avec le code anomalie MQRC\_BUFFER\_LENGTH\_ERROR.
- Le paramètre **Buffer** est déclaré comme étant de type String. Si les données à placer dans la file d'attente ne sont pas de type String, utilisez laAppel MQPUTAny à la place de MQPUT.

L'appel MQPUTAny possède les mêmes paramètres que l'appel MQPUT, sauf que le paramètre **Buffer** est déclaré comme étant de type Any, ce qui permet de placer n'importe quel type de données dans la file d'attente. Toutefois, cela signifie que Buffer ne peut pas être vérifié pour s'assurer qu'il a une taille d'au moins BufferLength octets.

## Appel C

```
MQPUT (Hconn, Hobj, &MsgDesc, &PutMsgOpts, BufferLength, Buffer,
      &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;          /* Connection handle */
MQHOBJ   Hobj;          /* Object handle */
MQMD     MsgDesc;       /* Message descriptor */
MQPMO    PutMsgOpts;    /* Options that control the action of MQPUT */
MQLONG   BufferLength;  /* Length of the message in Buffer */
MQBYTE   Buffer[n];     /* Message data */
MQLONG   CompCode;     /* Completion code */
MQLONG   Reason;       /* Reason code qualifying CompCode */
```

## Appel COBOL

```
CALL 'MQPUT' USING HCONN, HOBJ, MSGDESC, PUTMSGOPTS, BUFFERLENGTH,
                  BUFFER, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle
01 HCONN          PIC S9(9) BINARY.
** Object handle
01 HOBJ           PIC S9(9) BINARY.
** Message descriptor
01 MSGDESC.
   COPY CMQMDV.
** Options that control the action of MQPUT
01 PUTMSGOPTS.
   COPY CMQPMOV.
** Length of the message in BUFFER
01 BUFFERLENGTH  PIC S9(9) BINARY.
** Message data
01 BUFFER        PIC X(n).
** Completion code
01 COMPCODE      PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON        PIC S9(9) BINARY.
```

## Appel PL/I

```
call MQPUT (Hconn, Hobj, MsgDesc, PutMsgOpts, BufferLength, Buffer,  
            CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
dcl Hconn          fixed bin(31); /* Connection handle */  
dcl Hobj           fixed bin(31); /* Object handle */  
dcl MsgDesc        like MQMD;    /* Message descriptor */  
dcl PutMsgOpts     like MQPMO;    /* Options that control the action of  
                                MQPUT */  
dcl BufferLength    fixed bin(31); /* Length of the message in Buffer */  
dcl Buffer          char(n);      /* Message data */  
dcl CompCode       fixed bin(31); /* Completion code */  
dcl Reason         fixed bin(31); /* Reason code qualifying CompCode */
```

## Appel d'assembleur de haut niveau

```
CALL MQPUT, (HCONN,HOBJ,MSGDESC,PUTMSGOPTS,BUFFERLENGTH, X  
            BUFFER,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

HCONN	DS	F	Connection handle
HOBJ	DS	F	Object handle
MSGDESC	CMQMDA	,	Message descriptor
PUTMSGOPTS	CMQPMOA	,	Options that control the action of MQPUT
BUFFERLENGTH	DS	F	Length of the message in BUFFER
BUFFER	DS	CL(n)	Message data
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## Appel Visual Basic

Windows

```
MQPUT Hconn, Hobj, MsgDesc, PutMsgOpts, BufferLength, Buffer, CompCode,  
      Reason
```

Déclarez les paramètres comme suit :

```
Dim Hconn          As Long 'Connection handle'  
Dim Hobj           As Long 'Object handle'  
Dim MsgDesc        As MQMD 'Message descriptor'  
Dim PutMsgOpts     As MQPMO 'Options that control the action of MQPUT'  
Dim BufferLength    As Long 'Length of the message in Buffer'  
Dim Buffer          As String 'Message data'  
Dim CompCode       As Long 'Completion code'  
Dim Reason         As Long 'Reason code qualifying CompCode'
```

## MQPUT1 -Message d'insertion unique

L'appel MQPUT1 place un message dans une file d'attente, une liste de distribution ou une rubrique.

Il n'est pas nécessaire d'ouvrir la file d'attente, la liste de distribution ou la rubrique.

### Syntaxe


MQPUT1 (*Hconn, ObjDesc, MsgDesc, PutMsgOpts, BufferLength, Buffer, CompCode, Reason*)

## Paramètres

### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

 Sur les applications z/OS for CICS , l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour *Hconn*:

### MQHC\_DEF\_HCONN

Descripteur de connexion par défaut.

### ObjDesc

Type: MQOD-entrée/sortie

Il s'agit d'une structure qui identifie la file d'attente à laquelle le message est ajouté ou la rubrique à laquelle le message est publié. Voir «[MQOD-Descripteur d'objet](#)», à la page 501 pour des détails.

Si la structure est une file d'attente, l'utilisateur doit être autorisé à ouvrir la file d'attente pour la sortie. La file d'attente ne doit pas être une file d'attente modèle.

### MsgDesc

Type : MQMD - entrée/sortie

Cette structure décrit les attributs du message envoyé et reçoit des informations en retour une fois la demande d'insertion terminée. Voir «[MQMD-Descripteur de message](#)», à la page 438 pour des détails.

Si l'application fournit un MQMD version-1 , les données de message peuvent être préfixées avec une structure MQMDE afin de spécifier des valeurs pour les zones qui existent dans le MQMD version-2 mais pas dans version-1. Définissez la zone Format dans le MQMD sur MQFMT\_MD\_EXTENSION pour indiquer qu'un MQMDE est présent. Pour plus d'informations, voir «[MQMDE-Extension de descripteur de message](#)», à la page 492.

L'application n'a pas besoin de fournir une structure MQMD si un descripteur de message valide est fourni dans la zone MsgHandle de la structure MQGMO ou dans les zones OriginalMsgHandle ou NewMsgHandle de la structure MQPMO. Si aucune donnée n'est fournie dans l'un de ces champs, le descripteur du message est extrait du descripteur associé aux descripteurs de message.

### PutMsgOpts

Type: MQPMO-entrée/sortie

Pour plus d'informations, voir «[MQPMO-Options de message d'insertion](#)», à la page 522.

### BufferLength

Type : MQLONG - entrée

Longueur du message dans Buffer. La valeur zéro est valide et indique que le message ne contient aucune donnée d'application. La limite supérieure dépend de différents facteurs ; voir «[MQPUT-Insertion d'un message](#)», à la page 785 pour la description du paramètre **BufferLength** .

### Mémoire tampon

Type: MQBYTEXBufferLength-input

Il s'agit d'une mémoire tampon contenant les données de message d'application à envoyer. Alignez la mémoire tampon sur une limite appropriée à la nature des données dans le message. L'alignement sur 4 octets convient à la plupart des messages (y compris les messages contenant des structures d'en-tête IBM MQ ), mais certains messages peuvent nécessiter un alignement plus strict. Par exemple, un message contenant un entier binaire de 64 bits peut nécessiter un alignement de 8 octets.

Si Buffer contient des données de type caractère ou numérique, définissez les zones CodedCharSetId et Encoding du paramètre **MsgDesc** sur les valeurs appropriées aux données ; cela permet au destinataire du message de convertir les données (si nécessaire) en jeu de caractères et en codage utilisés par le destinataire.



**Remarque :** Tous les autres paramètres de l'appel MQPUT1 doivent être dans le jeu de caractères et le codage du gestionnaire de files d'attente local (indiqués par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et MQENC\_NATIVE).

Dans le langage de programmation C, le paramètre est déclaré comme un pointeur vers vide ; l'adresse de tout type de données peut être spécifiée comme paramètre.

Si la valeur du paramètre **BufferLength** est zéro, il n'est pas fait référence à **Buffer** ; dans ce cas, l'adresse de paramètre transmise par les programmes écrits en assembleur C ou System/390 peut être null.

### **CompCode**

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_WARNING**

Avertissement (achèvement partiel).

#### **MQCC\_FAILED**

Echec de l'appel.

### **Motif**

Type : MQLONG - sortie

Code anomalie qualifiant CompCode.

Si CompCode est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si CompCode est MQCC\_WARNING:

#### **MQRC\_MULTIPLE\_MOTIFS**

(2136, X'858') Plusieurs codes anomalie ont été renvoyés.

#### **MQRC\_INCOMPLETE\_GROUP**

(2241, X'8C1') Groupe de messages non complet.

#### **MQRC\_INCOMPLETE\_MSG**

(2242, X'8C2') Message logique non complet.

#### **MQRC\_PRIORITY\_MAXIMAL DÉPASSEMENTS**

(2049, X'801') La priorité de message dépasse la valeur maximale prise en charge.

#### **MQRC\_UNKNOWN\_REPORT\_OPTION**

(2104, X'838') Options de rapport dans le descripteur de message non reconnues.

Si CompCode est MQCC\_FAILED:

#### **MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') Adaptateur non disponible.

#### **MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

#### **MQRC\_ALIAS\_BASE\_Q\_TYPE\_ERROR**

(2001, X'7D1') Le type de la file d'attente de base de l'alias n'est pas valide.

#### **MQRC\_API\_EXIT\_ERROR**

(2374, X'946') Echec de l'exit API.

#### **MQRC\_API\_EXIT\_LOAD\_ERROR**

(2183, X'887') Impossible de charger l'exit API.

#### **MQRC\_ASID\_MISMATCH**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

**MQRC\_BACKED\_OUT**

(2003, X'7D3') Unité d'oeuvre annulée.

**MQRC\_BUFFER\_ERROR**

(2004, X'7D4') Paramètre de tampon non valide.

**MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

**MQRC\_CF\_NOT\_XX\_ENCODE\_CASE\_ONE disponible**

(2345, X' 929') unité de couplage non disponible.

**Echec de MQRC\_CF\_STRUC\_AUTH\_FAILED**

(2348, X'92C') La vérification d'autorisation de la structure d'unité de couplage a échoué.

**MQRC\_CF\_STRUC\_ERREUR**

(2349, X'92D') Structure d'unité de couplage non valide.

**MQRC\_CF\_STRUC\_FAILED**

(2373, X'945') Echec de la structure de l'unité de couplage.

**MQRC\_CF\_STRUC\_IN\_USE**

(2346, X'92A') Structure de l'unité de couplage en cours d'utilisation.

**MQRC\_CF\_STRUC\_LIST\_HDR\_IN\_USE**

(2347, X'92B') En-tête de liste de la structure d'unité de couplage en cours d'utilisation.

**MQRC\_CFGP\_ERREUR**

(2416, X' 970') La structure de paramètre de groupe PCF MQCFGR dans les données de message n'est pas valide.

**MQRC\_CFH\_ERREUR**

(2235, X'8BB') Structure d'en-tête PCF non valide.

**MQRC\_CFIF\_ERREUR**

(2414, X'96E') La structure du paramètre de filtre d'entier PCF dans les données de message n'est pas valide.

**MQRC\_CFIL\_ERREUR**

(2236, X'8BC') Structure de paramètre de liste d'entiers PCF ou PCIF\*64 structure de paramètre de liste d'entiers non valide.

**MQRC\_CFIN\_ERREUR**

(2237, X'8BD') Structure de paramètre d'entier PCF ou PCIF\*64 structure de paramètre d'entier non valide.

**MQRC\_CFSF\_ERREUR**

(2415, X'96F') La structure de paramètre de filtre de chaîne PCF dans les données de message n'est pas valide.

**MQRC\_CFSL\_ERREUR**

(2238, X'8BE') Structure de paramètre de liste de chaînes PCF incorrecte.

**MQRC\_CFST\_ERREUR**

(2239, X'8BF') Structure de paramètre de chaîne PCF incorrecte.

**MQRC\_CICS\_WAIT\_FAILED**

(2140, X'85C') Demande d'attente rejetée par CICS.

**MQRC\_CLUSTER\_EXIT\_ERREUR**

(2266, X'8DA') L'exit de charge de travail de cluster a échoué.

**Erreur de résolution MQRC\_CLUSTER\_RESOLUTION\_ERROR**

(2189, X'88D') La résolution du nom de cluster a échoué.

**MQRC\_CLUSTER\_RESOURCE\_ERROR**

(2269, X'8DD') Erreur de ressource de cluster.

**MQRC\_COD\_NOT\_VALID\_FOR\_XCF\_Q**  
(2106, X'83A') L'option de rapport COD n'est pas valide pour la file d'attente XCF.

**MQRC\_CONNECTION\_BROKEN**  
(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**MQRC\_CONNECTION\_NOT\_AUTHORIZED**  
(2217, X'8A9') Non autorisé pour la connexion.

**MQRC\_CONNECTION QUIESCING**  
(2202, X'89A') Connexion en cours de mise au repos.

**MQRC\_CONNECTION\_STOPPING**  
(2203, X'89B') Connexion en cours d'arrêt.

**MQRC\_CONTENT\_ERREUR**  
2554 (X'09FA') Le contenu du message n'a pas pu être analysé pour déterminer si le message peut être distribué à un abonné avec un sélecteur de message étendu.

**MQRC\_CONTEXT\_HANDLE\_ERREUR**  
(2097, X'831') L'identificateur de file d'attente auquel il est fait référence ne sauvegarde pas le contexte.

**MQRC\_CONTEXT\_NOT\_AVAILABLE**  
(2098, X'832') Contexte non disponible pour le descripteur de file d'attente référencé.

**MQRC\_DATA\_LENGTH\_ERROR**  
(2010, X'7DA') Paramètre de longueur des données non valide.

**MQRC\_DB2\_NOT\_AVAILABLE**  
(2342, X' 926') Sous-système Db2 non disponible.

**MQRC\_DEF\_XMIT\_Q\_TYPE\_ERREUR**  
(2198, X'896') La file d'attente de transmission par défaut n'est pas locale.

**MQRC\_DEF\_XMIT\_Q\_USAGE\_ERROR**  
(2199, X'897') Erreur d'utilisation de la file d'attente de transmission par défaut.

**MQRC\_DH\_ERREUR**  
(2135, X'857') Structure d'en-tête de distribution incorrecte.

**MQRC\_DLH\_ERREUR**  
(2141, X'85D') Structure d'en-tête de rebut non valide.

**MQRC\_EPH\_ERREUR**  
(2420, X' 974') Structure PCF intégrée incorrecte.

**MQRC\_EXPIRY\_ERROR**  
(2013, X'7DD') Heure d'expiration non valide.

**MQRC\_ERREUR\_FEEDBACK\_ERROR**  
(2014, X'7DE') Code retour non valide.

**MQRC\_GLOBAL\_UOW\_CONFLICT**  
(2351, X'92F') Conflit entre les unités d'oeuvre globales.

**MQRC\_GROUP\_ID\_ERREUR**  
(2258, X'8D2') Identificateur de groupe non valide.

**MQRC\_HANDLE\_IN\_USE\_FOR\_UOW**  
(2353, X'931') Descripteur en cours d'utilisation pour l'unité d'oeuvre globale.

**MQRC\_HANDLE\_NOT\_AVAILABLE**  
(2017, X'7E1') Plus de descripteurs disponibles.

**MQRC\_HCONN\_ERROR**  
(2018, X'7E2') Descripteur de connexion non valide.

**MQRC\_HEADER\_ERREUR**  
(2142, X'85E') La structure d'en-tête IBM MQ n'est pas valide.

**MQRC\_IIH\_ERREUR**  
(2148, X'864') La structure d'en-tête d'information IMS n'est pas valide.

**MQRC\_LOCAL\_UOW\_CONFLICT**

(2352, X'930') Conflit entre l'unité d'oeuvre globale et l'unité d'oeuvre locale.

**MQRC\_MD\_ERROR**

(2026, X'7EA') Descripteur de message non valide.

**MQRC\_MDE\_ERREUR**

(2248, X'8C8') Extension de descripteur de message non valide.

**MQRC\_MISSING\_REPLY\_TO\_Q**

(2027, X'7EB') File d'attente de réponse manquante.

**MQRC\_WIH**

(2332, X'91C') Les données de message ne commencent pas par MQWIH.

**MQRC\_MSG\_FLAGS\_ERREUR**

(2249, X'8C9') Indicateurs de message non valides.

**MQRC\_MSG\_SEQ\_NUMBER\_ERROR**

(2250, X'8CA') Numéro de séquence de message non valide.

**MQRC\_MSG\_TOO\_BIG\_FOR\_Q**

(2030, X'7EE') Longueur de message supérieure au maximum pour la file d'attente.

**MQRC\_MSG\_TOO\_BIG\_FOR\_Q\_MGR**

(2031, X'7EF') Longueur de message supérieure à la longueur maximale pour le gestionnaire de files d'attente.

**MQRC\_MSG\_TYPE\_ERREUR**

(2029, X'7ED') Le type de message dans le descripteur de message n'est pas valide.

**MQRC\_MULTIPLE\_MOTIFS**

(2136, X'858') Plusieurs codes anomalie ont été renvoyés.

**MQRC\_NO\_DESTINATIONS\_AVAILABLE**

(2270, X'8DE') Aucune file d'attente de destination disponible.

**MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') Non autorisé pour l'accès.

**MQRC\_OBJECT\_DAMAGED**

(2101, X'835') Objet endommagé.

**MQRC\_OBJECT\_IN\_USE**

(2042, X'7FA') Objet déjà ouvert avec des options en conflit.

**MQRC\_OBJET\_NIVEAU\_INCOMPATIBLE**

(2360, X'938') Niveau d'objet non compatible.

**MQRC\_OBJECT\_NAME\_ERROR**

(2152, X'868') Nom d'objet incorrect.

**MQRC\_OBJECT\_NOT\_UNIQUE**

(2343, X'927') Objet non unique.

**MQRC\_OBJECT\_Q\_MGR\_NAME\_ERROR**

(2153, X'869') Nom de gestionnaire de files d'attente d'objet incorrect.

**MQRC\_OBJECT\_RECORDS\_ERROR**

(2155, X'86B') Enregistrements d'objet non valides.

**MQRC\_OBJECT\_TYPE\_ERREUR**

(2043, X'7FB') Type d'objet non valide.

**MQRC\_ERREUR D'OD\_DU**

(2044, X'7FC') Structure de descripteur d'objet non valide.

**ERREUR MQRC\_OFFSET\_ERROR**

(2251, X'8CB') Position de segment de message non valide.

**MQRC\_OPTIONS\_ERROR**

(2046, X'7FE') Options incorrectes ou incohérentes.

**MQRC\_ORIGINAL\_LENGTH\_ERROR**  
(2252, X'8CC') Longueur d'origine incorrecte.

**MQRC\_PAGESET\_ERROR**  
(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.

**MQRC\_PAGESET\_FULL**  
(2192, X'890') Support de stockage externe saturé.

**MQRC\_PCF\_ERREUR**  
(2149, X'865') Structures PCF incorrectes.

**MQRC\_XX\_ENCODE\_CASE\_ONE err\_persistent**  
(2047, X'7FF') Persistance non valide.

**MQRC\_PERSISTENT\_NOT\_ALLOWED**  
(2048, X'800') La file d'attente ne prend pas en charge les messages persistants.

**MQRC\_PMO\_ERREUR**  
(2173, X'87D') La structure des options d'insertion de message n'est pas valide.

**MQRC\_PMO\_RECORD\_FLAGS\_ERROR**  
(2158, X'86E') Les indicateurs d'enregistrement de message d'insertion ne sont pas valides.

**MQRC\_PRIORITY\_ERROR**  
(2050, X'802') Priorité de message non valide.

**MQRC\_PUBLICATION\_FAILURE**  
(2502, X'9C6') La publication n'a été distribuée à aucun des abonnés.

**MQRC\_PUT\_INHIBÉ**  
(2051, X'803') Appels d'insertion bloqués pour la file d'attente.

**MQRC\_PUT\_MSG\_RECORDS\_ERROR**  
(2159, X'86F') Enregistrements de message d'insertion non valides.

**MQRC\_Q\_DELETED**  
(2052, X'804') File d'attente supprimée.

**MQRC\_Q\_FULL**  
(2053, X'805') La file d'attente contient déjà le nombre maximal de messages.

**MQRC\_Q\_MGR\_NAME\_ERROR**  
(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**  
(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

**MQRC\_Q\_MGR QUIESCING**  
(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.

**MQRC\_Q\_MGR\_STOPPING**  
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**MQRC\_Q ESPACE\_NON\_DISPONIBLE**  
(2056, X'808') Aucun espace disponible sur le disque pour la file d'attente.

**MQRC\_Q\_TYPE\_ERREUR**  
(2057, X'809') Type de file d'attente non valide.

**MQRC\_RECS\_Présentateur-ERREUR**  
(2154, X'86A') Nombre d'enregistrements présents non valide.

**MQRC\_REMOTE\_Q\_NAME\_ERROR**  
(2184, X'888') Nom de file d'attente éloignée incorrect.

**MQRC\_REPORT\_OPTIONS\_ERREUR**  
(2061, X'80D') Les options de rapport dans le descripteur de message ne sont pas valides.

**MQRC\_RESOURCE\_PROBLEM**  
(2102, X'836') Ressources système disponibles insuffisantes.

**MQRC\_RESPONSE\_RECORDS\_ERROR**  
(2156, X'86C') Enregistrements de réponse non valides.

**MQRC\_RFH\_ERREUR**  
(2334, X'91E') Structure MQRFH ou MQRFH2 non valide.

**MQRC\_RMH\_ERREUR**  
(2220, X'8AC') La structure d'en-tête de message de référence n'est pas valide.

**MQRC\_SECURITY\_ERROR (ERREUR DE SECURITE MQ)**  
(2063, X'80F') Une erreur de sécurité s'est produite.

**MQRC\_SEGMENT\_LONGUEUR\_ZÉRO**  
(2253, X'8CD') La longueur des données dans le segment de message est égale à zéro.

**MQRC\_SELECTION\_NOT\_AVAILABLE**  
2551 (X'09F7') Il existe un abonné possible pour la publication, mais le gestionnaire de files d'attente ne peut pas vérifier si la publication doit être envoyée à l'abonné.

**MQRC\_STOPPED\_BY\_CLUSTER\_EXIT**  
(2188, X'88C') Appel rejeté par l'exit de charge de travail du cluster.

**MQRC\_STORAGE\_CLASS\_ERROR**  
(2105, X'839') Erreur de classe de stockage.

**MQRC\_STORAGE\_MEDIUM\_FULL**  
(2192, X'890') Support de stockage externe saturé.

**MQRC\_STORAGE\_NOT\_AVAILABLE**  
(2071, X'817') Mémoire disponible insuffisante.

**MQRC\_SUPPRESSED\_BY\_EXIT**  
(2109, X'83D') Appel supprimé par le programme d'exit.

**MQRC\_SYNCPOINT\_LIMIT\_REACHED**  
(2024, X'7E8') Aucun autre message pouvant être traité au sein de l'unité d'oeuvre en cours.

**MQRC\_SYNCPOINT\_NOT\_AVAILABLE**  
(2072, X'818') Prise en charge du point de synchronisation non disponible.

**MQRC\_TM\_ERREUR**  
(2265, X'8D9') La structure du message de déclenchement n'est pas valide.

**MQRC\_TMC\_ERREUR**  
(2191, X'88F') La structure de message du déclencheur de caractères n'est pas valide.

**MQRC\_UNEXPECTED\_ERROR**  
(2195, X'893') Une erreur inattendue s'est produite.

**MQRC\_UNKNOWN\_ALIAS\_BASE\_Q**  
(2082, X'822') File d'attente de base alias inconnue.

**MQRC\_UNKNOWN\_DEF\_XMIT\_Q**  
(2197, X'895') File d'attente de transmission par défaut inconnue.

**MQRC\_UNKNOWN\_NOM\_OBJET**  
(2085, X'825') Nom d'objet inconnu.

**MQRC\_UNKNOWN\_OBJET\_Q\_MGR**  
(2086, X'826') Gestionnaire de files d'attente d'objets inconnu.

**MQRC\_UNKNOWN\_REMOTE\_Q\_MGR**  
(2087, X'827') Gestionnaire de files d'attente éloignées inconnu.

**MQRC\_UNKNOWN\_XMIT\_Q**  
(2196, X'894') File d'attente de transmission inconnue.

**MQRC\_UOW\_ENLISTMENT\_ERROR**  
(2354, X'932') Echec de l'inscription dans l'unité d'oeuvre globale.

**MQRC\_UOW\_MIX\_NOT\_SUPPORTED**  
(2355, X'933') Mélange d'appels d'unité d'oeuvre non pris en charge.

**MQRC\_UOW\_NOT\_AVAILABLE**  
(2255, X'8CF') Unité d'oeuvre non disponible pour le gestionnaire de files d'attente à utiliser.

**MQRC\_WIH\_ERREUR**

(2333, X'91D') Structure MQWIH non valide.

**MQRC\_WRONG\_CF\_NIVEAU**

(2366, X'93E') La structure de l'unité de couplage est de niveau incorrect.

**MQRC\_WRONG\_MD\_VERSION**

(2257, X'8D1') Version MQMD fournie non valide.

**MQRC\_XMIT\_Q\_TYPE\_ERROR**

(2091, X'82B') File d'attente de transmission non locale.

**MQRC\_XMIT\_Q\_USAGE\_ERROR**

(2092, X'82C') File d'attente de transmission avec une utilisation incorrecte.

**MQRC\_XQH\_ERREUR**

(2260, X'8D4') Structure d'en-tête de file d'attente de transmission non valide.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Remarques sur l'utilisation

1. Les appels MQPUT et MQPUT1 peuvent être utilisés pour placer des messages dans une file d'attente ; l'appel à utiliser dépend des circonstances:
  - Utilisez l'appel MQPUT pour placer plusieurs messages dans la *même* file d'attente.  
Un appel MQOPEN spécifiant l'option MQOO\_OUTPUT est émis en premier, suivi d'une ou de plusieurs demandes MQPUT pour ajouter des messages à la file d'attente ; enfin, la file d'attente est fermée avec un appel MQCLOSE. Cela offre de meilleures performances que l'utilisation répétée de l'appel MQPUT1 .
  - Utilisez l'appel MQPUT1 pour placer *un seul* message dans une file d'attente.  
Cet appel encapsule les appels MQOPEN, MQPUT et MQCLOSE en un seul appel, ce qui réduit le nombre d'appels qui doivent être émis.
2. Si une application place une séquence de messages dans la même file d'attente sans utiliser de groupes de messages, l'ordre de ces messages est préservé sous réserve que certaines conditions soient satisfaites. Toutefois, dans la plupart des environnements, l'appel MQPUT1 ne répond pas à ces conditions et ne préserve donc pas l'ordre des messages. L'appel MQPUT doit être utilisé à la place dans ces environnements. Pour plus de détails, voir [Remarques sur l'utilisation de MQPUT](#).
3. L'appel MQPUT1 peut être utilisé pour insérer des messages dans des listes de distribution. Pour des informations générales à ce sujet, voir les remarques sur la syntaxe des appels MQOPEN et MQPUT.

Les listes de distribution sont prises en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour les clients IBM MQ connectés à ces systèmes.

Les différences suivantes s'appliquent lors de l'utilisation de l'appel MQPUT1 :

- a. Si l'application fournit des enregistrements de réponse MQRR, ils doivent être fournis à l'aide de la structure MQOD ; ils ne peuvent pas être fournis à l'aide de la structure MQPMO.
- b. Le code anomalie MQRC\_OPEN\_FAILED n'est jamais renvoyé par MQPUT1 dans les enregistrements de réponse ; si l'ouverture d'une file d'attente échoue, l'enregistrement de réponse de cette file d'attente contient le code anomalie résultant de l'opération d'ouverture.

Si une opération d'ouverture d'une file d'attente aboutit avec le code achèvement MQCC\_WARNING, le code achèvement et le code anomalie de l'enregistrement de réponse de

cette file d'attente sont remplacés par les codes achèvement et anomalie résultant de l'opération d'insertion.

Comme pour les appels MQOPEN et MQPUT, le gestionnaire de files d'attente définit les enregistrements de réponse (s'ils sont fournis) uniquement lorsque le résultat de l'appel n'est pas le même pour toutes les files d'attente de la liste de distribution ; ceci est indiqué par l'exécution de l'appel avec le code anomalie MQRC\_MULTIPLE\_MOTIFS.

4. Si l'appel MQPUT1 est utilisé pour insérer un message dans une file d'attente de cluster, l'appel se comporte comme si MQOO\_BIND\_NOT\_FIXED avait été spécifié sur l'appel MQOPEN.
5. Si un message est inséré avec une ou plusieurs structures d'en-tête IBM MQ au début des données de message d'application, le gestionnaire de files d'attente effectue certaines vérifications sur les structures d'en-tête pour vérifier qu'elles sont valides. Pour plus d'informations à ce sujet, voir les remarques sur l'utilisation de l'appel MQPUT.
6. Si plusieurs situations d'avertissement se produisent (voir le paramètre **CompCode** ), le code anomalie renvoyé est le premier de la liste suivante qui s'applique:
  - a. MQRC\_MULTIPLE\_MOTIFS
  - b. MQRC\_INCOMPLETE\_MSG
  - c. MQRC\_INCOMPLETE\_GROUP
  - d. MQRC\_PRIORITY\_DÉPASSS\_MAXIMUM ou MQRC\_UNKNOWN\_REPORT\_OPTION
7. Pour le langage de programmation Visual Basic, les points suivants s'appliquent :
  - Si la taille du paramètre **Buffer** est inférieure à la longueur spécifiée par le paramètre **BufferLength** , l'appel échoue avec le code anomalie MQRC\_BUFFER\_LENGTH\_ERROR.
  - Le paramètre **Buffer** est déclaré comme étant de type String. Si les données à placer dans la file d'attente ne sont pas de type String, utilisez laMQPUT1Any à la place de MQPUT1.L'appel MQPUT1Any possède les mêmes paramètres que l'appel MQPUT1 , sauf que le paramètre **Buffer** est déclaré comme étant de type Any, ce qui permet de placer n'importe quel type de données dans la file d'attente. Toutefois, cela signifie que Buffer ne peut pas être vérifié pour s'assurer qu'il a une taille d'au moins BufferLength octets.
8. Lorsqu'un appel MQPUT1 est émis avec MQPMO\_SYNCPOINT, le comportement par défaut change, de sorte que l'opération d'insertion est exécutée de manière asynchrone. Cela peut entraîner une modification du comportement de certaines applications qui s'appuient sur certaines zones des structures MQOD et MQMD renvoyées, mais qui contiennent désormais des valeurs non définies. Une application peut spécifier MQPMO\_SYNC\_RESPONSE pour s'assurer que l'opération d'insertion est effectuée de manière synchrone et que toutes les valeurs de zone appropriées sont terminées.

## Appel C

```
MQPUT1 (Hconn, &ObjDesc, &MsgDesc, &PutMsgOpts,  
        BufferLength, Buffer, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;           /* Connection handle */  
MQOD     ObjDesc;        /* Object descriptor */  
MQMD     MsgDesc;        /* Message descriptor */  
MQPMO    PutMsgOpts;     /* Options that control the action of MQPUT1 */  
MQLONG   BufferLength;    /* Length of the message in Buffer */  
MQBYTE   Buffer[n];      /* Message data */  
MQLONG   CompCode;       /* Completion code */  
MQLONG   Reason;        /* Reason code qualifying CompCode */
```



## Appel COBOL

```
CALL 'MQPUT1' USING HCONN, OBJDESC, MSGDESC, PUTMSGOPTS,  
                   BUFFERLENGTH, BUFFER, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle  
01 HCONN          PIC S9(9) BINARY.  
** Object descriptor  
01 OBJDESC.  
   COPY CMQODV.  
** Message descriptor  
01 MSGDESC.  
   COPY CMQMDV.  
** Options that control the action of MQPUT1  
01 PUTMSGOPTS.  
   COPY CMQPMOV.  
** Length of the message in BUFFER  
01 BUFFERLENGTH PIC S9(9) BINARY.  
** Message data  
01 BUFFER        PIC X(n).  
** Completion code  
01 COMPCODE      PIC S9(9) BINARY.  
** Reason code qualifying COMPCODE  
01 REASON        PIC S9(9) BINARY.
```

## Appel PL/I

```
call MQPUT1 (Hconn, ObjDesc, MsgDesc, PutMsgOpts, BufferLength, Buffer,  
            CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
dcl Hconn          fixed bin(31); /* Connection handle */  
dcl ObjDesc       like MQOD;    /* Object descriptor */  
dcl MsgDesc       like MQMD;    /* Message descriptor */  
dcl PutMsgOpts    like MQPMO;   /* Options that control the action of  
                                MQPUT1 */  
dcl BufferLength   fixed bin(31); /* Length of the message in Buffer */  
dcl Buffer         char(n);      /* Message data */  
dcl CompCode      fixed bin(31); /* Completion code */  
dcl Reason        fixed bin(31); /* Reason code qualifying CompCode */
```

## Appel d'assembleur de haut niveau

```
CALL MQPUT1, (HCONN,OBJDESC,MSGDESC,PUTMSGOPTS,BUFFERLENGTH, X  
             BUFFER,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
HCONN          DS      F      Connection handle  
OBJDESC        CMQODA   ,      Object descriptor  
MSGDESC        CMQMDA   ,      Message descriptor  
PUTMSGOPTS     CMQPMOA  ,      Options that control the action of MQPUT1  
BUFFERLENGTH   DS      F      Length of the message in BUFFER  
BUFFER         DS      CL(n)  Message data  
COMPCODE       DS      F      Completion code  
REASON         DS      F      Reason code qualifying COMPCODE
```

## Appel Visual Basic

Windows

```
MQPUT1 Hconn, ObjDesc, MsgDesc, PutMsgOpts, BufferLength, Buffer,
      CompCode, Reason
```

Déclarez les paramètres comme suit :

```
Dim Hconn      As Long   'Connection handle'
Dim ObjDesc    As MQOD   'Object descriptor'
Dim MsgDesc    As MQMD   'Message descriptor'
Dim PutMsgOpts As MQPMO  'Options that control the action of MQPUT1'
Dim BufferLength As Long  'Length of the message in Buffer'
Dim Buffer      As String 'Message data'
Dim CompCode   As Long   'Completion code'
Dim Reason     As Long   'Reason code qualifying CompCode'
```

## MQSET-Définition des attributs d'objet

Utilisez l'appel MQSET pour modifier les attributs d'un objet représenté par un descripteur. L'objet doit être une file d'attente.

### Syntaxe


MQSET (*Hconn, Hobj, SelectorCount, Sélecteurs, IntAttrCount, IntAttrs, CharAttrLength, CharAttrs, Compcode, Motif*)

### Paramètres

#### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de Hconn a été renvoyée par un appel MQCONN ou MQCONNX précédent.

 Sur les applications z/OS for CICS , l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour Hconn:

#### MQHC\_DEF\_HCONN

Descripteur de connexion par défaut.

#### HOBJ

Type : MQHOBJ - entrée

Cet identificateur représente l'objet de file d'attente avec les attributs à définir. Le descripteur a été renvoyé par un appel MQOPEN précédent qui a spécifié l'option MQOO\_SET.

#### SelectorCount

Type : MQLONG - entrée

Nombre de sélecteurs fournis dans le tableau *Selectors* . Il s'agit du nombre d'attributs à définir. Zéro est une valeur valide. Le nombre maximal autorisé est 256.

#### Sélecteurs

Type: MQLONGxSelectorNombre-entrée

Il s'agit d'un tableau de sélecteurs d'attribut **SelectorCount** ; chaque sélecteur identifie un attribut (entier ou caractère) avec une valeur à définir.

Chaque sélecteur doit être valide pour le type de file d'attente représenté par Hobj . Seules certaines valeurs MQIA\_\* et MQCA\_\* sont autorisées, comme indiqué ultérieurement.

Les sélecteurs peuvent être spécifiés dans n'importe quel ordre. Les valeurs d'attribut qui correspondent à des sélecteurs d'attribut d'entier (sélecteurs MQIA\_\*) doivent être spécifiées dans IntAttrs dans l'ordre dans lequel ces sélecteurs apparaissent dans Selectors. Les valeurs d'attribut qui correspondent à des sélecteurs d'attribut de caractères (sélecteurs MQCA\_\*) doivent être spécifiées dans CharAttrs dans l'ordre dans lequel ces sélecteurs apparaissent. Les sélecteurs

MQIA\_\* peuvent être imbriqués avec les sélecteurs MQCA\_\* ; seul l'ordre relatif au sein de chaque type est important.

Vous pouvez spécifier le même sélecteur plusieurs fois ; si vous le faites, la dernière valeur spécifiée pour un sélecteur particulier est celle qui prend effet.

**Remarque :**

1. Les sélecteurs d'attribut d'entier et de caractère sont alloués dans deux plages différentes ; les sélecteurs MQIA\_\* résident dans la plage MQIA\_FIRST à MQIA\_LAST et les sélecteurs MQCA\_\* dans la plage MQCA\_FIRST à MQCA\_LAST.

Pour chaque plage, les constantes MQIA\_LAST\_USED et MQCA\_LAST\_USED définissent la valeur la plus élevée acceptée par le gestionnaire de files d'attente.

2. Si tous les sélecteurs MQIA\_\* apparaissent en premier, les mêmes numéros d'élément peuvent être utilisés pour adresser les éléments correspondants dans les tableaux `Selectors` et `IntAttrs`.

3. Si la valeur du paramètre **SelectorCount** est zéro, il n'est pas fait référence à `Selectors` ; dans ce cas, l'adresse de paramètre transmise par les programmes écrits en assembleur C ou System/390 peut être null.

Les attributs qui peuvent être définis sont répertoriés dans le tableau suivant. Aucun autre attribut ne peut être défini à l'aide de cet appel. Pour les sélecteurs d'attribut MQCA\_\*, la constante qui définit la longueur en octets de la chaîne requise dans `CharAttrs` est fournie entre parenthèses.

Sélecteur	Description	Remarque
DONNEES_TRIGGER_MQCA	Données de déclenchement (MQ_TRIGGER_DATA_LENGTH).	
MQIA_DIST_LISTS	Prise en charge de la liste de distribution.	1
MQIA_INHIBIT_GET	Indique si les opérations d'extraction sont autorisées.	
MQIA_INHIBIT_PUT	Indique si les opérations d'insertion sont autorisées.	
MQIA_TRIGGER_CONTROL	Contrôle du déclencheur.	
PROFONDEUR-MQIA_TRIGGER_PROFONDEUR	Longueur de déclenchement.	
MQIA_TRIGGER_MSG_PRIORITY	Priorité de message de seuil pour les déclencheurs.	
TYPE_TRIGGER_MQIA_	Type de déclencheur.	

**Remarque :**

1. Pris en charge uniquement sur les plateformes suivantes:

-  AIX
-  IBM i
-  Linux
-  Windows

et pour IBM MQ MQI clients connecté à ces systèmes.

**IntAttrCount**

Type : MQLONG - entrée

Il s'agit du nombre d'éléments dans le tableau `IntAttrs` et doit être au moins égal au nombre de sélecteurs `MQIA_*` dans le paramètre **Selectors** . Zéro est une valeur valide s'il n'y en a pas.

### **IntAttrs**

Type: `MQLONGxIntAttrCount` -entrée

Il s'agit d'un tableau de valeurs d'attribut d'entier `IntAttrCount` . Ces valeurs d'attribut doivent être dans le même ordre que les sélecteurs `MQIA_*` dans le tableau `Selectors` .

Si le paramètre **IntAttrCount** ou **SelectorCount** est égal à zéro, `IntAttrs` n'est pas référencé ; dans ce cas, l'adresse de paramètre transmise par les programmes écrits en assembleur C ou System/390 peut être null.

### **CharAttrLongueur**

Type : `MQLONG` - entrée

Il s'agit de la longueur en octets du paramètre **CharAttrs** , qui doit être au moins la somme des longueurs des attributs de caractères spécifiés dans le tableau `Selectors` . Zéro est une valeur valide s'il n'existe pas de sélecteurs `MQCA_*` dans `Selectors`.

### **CharAttrs**

Type: `MQCHAR x CharAttrLongueur`-entrée

Il s'agit de la mémoire tampon contenant les valeurs d'attribut de caractère, concaténées. La longueur de la mémoire tampon est indiquée par le paramètre **CharAttrLength** .

Les attributs de caractères doivent être spécifiés dans le même ordre que les sélecteurs `MQCA_*` dans le tableau `Selectors` . La longueur de chaque attribut de caractère est fixe (voir Sélecteurs ). Si la valeur à définir pour un attribut contient moins de caractères non blancs que la longueur définie de l'attribut, remplissez la valeur dans `CharAttrs` à droite avec des blancs pour que la valeur de l'attribut corresponde à la longueur définie de l'attribut.

Si le paramètre **CharAttrLength** ou **SelectorCount** est égal à zéro, `CharAttrs` n'est pas référencé ; dans ce cas, l'adresse de paramètre transmise par les programmes écrits en assembleur C ou System/390 peut être null.

### **CompCode**

Type : `MQLONG` - sortie

Code achèvement ; il s'agit d'un des codes suivants :

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

### **Motif**

Type : `MQLONG` - sortie

Code anomalie qualifiant `CompCode`.

Si `CompCode` est `MQCC_OK`:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si `CompCode` est `MQCC_FAILED`:

#### **MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') Adaptateur non disponible.

#### **MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

#### **MQRC\_API\_EXIT\_ERROR**

(2374, X'946') Echec de l'exit API.

**MQRC\_API\_EXIT\_LOAD\_ERROR**  
(2183, X'887') Impossible de charger l'exit API.

**MQRC\_ASID\_MISMATCH**  
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

**MQRC\_CALL\_IN\_PROGRESS**  
(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

**MQRC\_CF\_NOT\_XX\_ENCODE\_CASE\_ONE disponible**  
(2345, X' 929') Unité de couplage non disponible.

**MQRC\_CF\_STRUC\_FAILED**  
(2373, X'945') Echec de la structure de l'unité de couplage.

**MQRC\_CF\_STRUC\_IN\_USE**  
(2346, X'92A') Structure de l'unité de couplage en cours d'utilisation.

**MQRC\_CF\_STRUC\_LIST\_HDR\_IN\_USE**  
(2347, X'92B') En-tête de liste de la structure d'unité de couplage en cours d'utilisation.

**MQRC\_CHAR\_ATTR\_LENGTH\_ERROR**  
(2006, X'7D6') Longueur des attributs de caractère non valide.

**MQRC\_CHAR\_ATTRS\_ERREUR**  
(2007, X'7D7') Chaîne d'attributs de caractères non valide.

**MQRC\_CICS\_WAIT\_FAILED**  
(2140, X'85C') Demande d'attente rejetée par CICS.

**MQRC\_CONNECTION\_BROKEN**  
(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**MQRC\_CONNECTION\_NOT\_AUTHORIZED**  
(2217, X'8A9') Non autorisé pour la connexion.

**MQRC\_CONNECTION\_STOPPING**  
(2203, X'89B') Connexion en cours d'arrêt.

**MQRC\_DB2\_NOT\_AVAILABLE**  
(2342, X' 926') Sous-système Db2 non disponible.

**MQRC\_HCONN\_ERROR**  
(2018, X'7E2') Descripteur de connexion non valide.

**MQRC\_HOBJ\_ERROR**  
(2019, X'7E3') Descripteur d'objet non valide.

**MQRC\_INHIBIT\_VALEUR\_ERREUR**  
(2020, X'7E4') Valeur de l'attribut de file d'attente d'interdiction d'extraction ou d'interdiction d'insertion non valide.

**MQRC\_INT\_ATTR\_COUNT\_ERROR**  
(2021, X'7E5') Nombre d'attributs de type entier non valide.

**MQRC\_INT\_ATTRS\_ARRAY\_ERREUR**  
(2023, X'7E7') Tableau d'attributs d'entier non valide.

**MQRC\_NOT\_OPEN\_FOR\_SET**  
(2040, X'7F8') File d'attente non ouverte pour l'ensemble.

**MQRC\_OBJECT\_CHANGED**  
(2041, X'7F9') Définition d'objet modifiée depuis son ouverture.

**MQRC\_OBJECT\_DAMAGED**  
(2101, X'835') Objet endommagé.

**MQRC\_PAGESET\_ERROR**  
(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.

**MQRC\_Q\_DELETED**  
(2052, X'804') File d'attente supprimée.

**MQRC\_Q\_MGR\_NAME\_ERROR**

(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**

(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') Ressources système disponibles insuffisantes.

**MQRC\_SELECTOR\_COUNT\_ERREUR**

(2065, X'811') Nombre de sélecteurs non valide.

**MQRC\_SELECTOR\_ERREUR**

(2067, X'813') Sélecteur d'attribut non valide.

**MQRC\_SELECTOR\_LIMIT\_EXCEEDED**

(2066, X'812') Nombre de sélecteurs trop grand.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') Mémoire disponible insuffisante.

**MQRC\_SUPPRESSED\_BY\_EXIT**

(2109, X'83D') Appel supprimé par le programme d'exit.

**MQRC\_TRIGGER\_CONTROL\_ERROR**

(2075, X'81B') Valeur de l'attribut trigger-control non valide.

**MQRC\_TRIGGER\_DEPTH\_ERREUR**

(2076, X'81C') Valeur de l'attribut trigger-depth non valide.

**MQRC\_TRIGGER\_MSG\_PRIORITY\_ERR**

(2077, X'81D') Valeur de l'attribut trigger-message-priority non valide.

**MQRC\_TRIGGER\_TYPE\_ERREUR**

(2078, X'81E') Valeur de l'attribut de type de déclencheur non valide.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Remarques sur l'utilisation

1. A l'aide de cet appel, l'application peut spécifier un tableau d'attributs entiers ou une collection de chaînes d'attributs de caractères, ou les deux. Si aucune erreur ne se produit, les attributs spécifiés sont tous définis simultanément. Si une erreur se produit (par exemple, si un sélecteur n'est pas valide ou qu'une tentative est effectuée pour définir un attribut sur une valeur non valide), l'appel échoue et aucun attribut n'est défini.
2. Les valeurs des attributs peuvent être déterminées à l'aide de l'appel MQINQ ; voir «[MQINQ-Attributs d'objet Inquire](#)», à la page 737 pour plus de détails.

**Remarque :** Les valeurs des attributs dont les valeurs peuvent être obtenues à l'aide de l'appel MQINQ ne peuvent pas toutes être modifiées à l'aide de l'appel MQSET. Par exemple, aucun attribut d'objet de processus ou de gestionnaire de files d'attente ne peut être défini avec cet appel.

3. Les modifications d'attribut sont conservées lors des redémarrages du gestionnaire de files d'attente (autres que les modifications apportées aux files d'attente dynamiques temporaires qui ne survivent pas aux redémarrages du gestionnaire de files d'attente).
4. Vous ne pouvez pas modifier les attributs d'une file d'attente modèle à l'aide de l'appel MQSET. Toutefois, si vous ouvrez une file d'attente modèle à l'aide de l'appel MQOPEN avec l'option MQOO\_SET, vous pouvez utiliser l'appel MQSET pour définir les attributs de la file d'attente locale dynamique créée par l'appel MQOPEN.
5. Si l'objet défini est une file d'attente de cluster, il doit y avoir une instance locale de la file d'attente de cluster pour que l'ouverture aboutisse.

Pour plus d'informations sur les attributs d'objet, voir:

- «Attributs des files d'attente», à la page 875
- «Attributs des listes de noms», à la page 911
- «Attributs des définitions de processus», à la page 913
- «Attributs du gestionnaire de files d'attente», à la page 835

## Appel C

```
MQSET (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs,  
CharAttrLength, CharAttrs, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;           /* Connection handle */  
MQHOBJ   Hobj;           /* Object handle */  
MQLONG   SelectorCount; /* Count of selectors */  
MQLONG   Selectors[n];  /* Array of attribute selectors */  
MQLONG   IntAttrCount;  /* Count of integer attributes */  
MQLONG   IntAttrs[n];   /* Array of integer attributes */  
MQLONG   CharAttrLength; /* Length of character attributes buffer */  
MQCHAR   CharAttrs[n];  /* Character attributes */  
MQLONG   CompCode;      /* Completion code */  
MQLONG   Reason;        /* Reason code qualifying CompCode */
```

## Appel COBOL

```
CALL 'MQSET' USING HCONN, HOBJ, SELECTORCOUNT, SELECTORS-TABLE,  
INTATTRCOUNT, INTATTRS-TABLE, CHARATTRLENGTH,  
CHARATTRS, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle  
01 HCONN          PIC S9(9) BINARY.  
** Object handle  
01 HOBJ          PIC S9(9) BINARY.  
** Count of selectors  
01 SELECTORCOUNT PIC S9(9) BINARY.  
** Array of attribute selectors  
01 SELECTORS-TABLE.  
02 SELECTORS     PIC S9(9) BINARY OCCURS n TIMES.  
** Count of integer attributes  
01 INTATTRCOUNT PIC S9(9) BINARY.  
** Array of integer attributes  
01 INTATTRS-TABLE.  
02 INTATTRS     PIC S9(9) BINARY OCCURS n TIMES.  
** Length of character attributes buffer  
01 CHARATTRLENGTH PIC S9(9) BINARY.  
** Character attributes  
01 CHARATTRS     PIC X(n).  
** Completion code  
01 COMPCODE      PIC S9(9) BINARY.  
** Reason code qualifying COMPCODE  
01 REASON        PIC S9(9) BINARY.
```

## Appel PL/I

```
call MQSET (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount,  
IntAttrs, CharAttrLength, CharAttrs, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```

dcl Hconn          fixed bin(31); /* Connection handle */
dcl Hobj           fixed bin(31); /* Object handle */
dcl SelectorCount  fixed bin(31); /* Count of selectors */
dcl Selectors(n)   fixed bin(31); /* Array of attribute selectors */
dcl IntAttrCount   fixed bin(31); /* Count of integer attributes */
dcl IntAttrs(n)    fixed bin(31); /* Array of integer attributes */
dcl CharAttrLength fixed bin(31); /* Length of character attributes
                                buffer */
dcl CharAttrs      char(n);       /* Character attributes */
dcl CompCode       fixed bin(31); /* Completion code */
dcl Reason         fixed bin(31); /* Reason code qualifying
                                CompCode */

```

## Appel d'assembleur de haut niveau

```

CALL MQSET, (HCONN, HOBJ, SELECTORCOUNT, SELECTORS, INTATTRCOUNT, X
            INTATTRS, CHARATTRLENGTH, CHARATTRS, COMPCODE, REASON)

```

Déclarez les paramètres comme suit :

HCONN	DS	F	Connection handle
HOBJ	DS	F	Object handle
SELECTORCOUNT	DS	F	Count of selectors
SELECTORS	DS	(n)F	Array of attribute selectors
INTATTRCOUNT	DS	F	Count of integer attributes
INTATTRS	DS	(n)F	Array of integer attributes
CHARATTRLENGTH	DS	F	Length of character attributes buffer
CHARATTRS	DS	CL(n)	Character attributes
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## Appel Visual Basic

```

MQSET Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs,
CharAttrLength, CharAttrs, CompCode, Reason

```

Déclarez les paramètres comme suit :

Dim Hconn	As Long	'Connection handle'
Dim Hobj	As Long	'Object handle'
Dim SelectorCount	As Long	'Count of selectors'
Dim Selectors	As Long	'Array of attribute selectors'
Dim IntAttrCount	As Long	'Count of integer attributes'
Dim IntAttrs	As Long	'Array of integer attributes'
Dim CharAttrLength	As Long	'Length of character attributes buffer'
Dim CharAttrs	As String	'Character attributes'
Dim CompCode	As Long	'Completion code'
Dim Reason	As Long	'Reason code qualifying CompCode'

## MQSETMP-Définition de la propriété de message

Utilisez l'appel MQSETMP pour définir ou modifier une propriété d'un descripteur de message.

### Syntaxe

```

MQSETMP (Hconn, Hmsg, SetPropOpts, Name, PropDesc, Type, ValueLength, Value, Compcode, Reason)

```

### Paramètres

#### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente.



La valeur doit correspondre au descripteur de connexion utilisé pour créer le descripteur de message spécifié dans le paramètre **Hmsg** . Si le descripteur de message a été créé à l'aide de MQHC\_UNASSOCIATED\_HCONN, une connexion valide doit être établie sur l'unité d'exécution définissant une propriété du descripteur de message, sinon l'appel échoue avec le code anomalie MQRC\_CONNECTION\_BROKEN.

### **Msg**

Type: MQHMSG-entrée

Il s'agit du descripteur de message à modifier. La valeur a été renvoyée par un appel MQCRTMH précédent.

### **Options SetProp**

Type: MQSMPO-entrée

Contrôle la façon dont les propriétés de message sont définies.

Cette structure permet aux applications de spécifier des options qui contrôlent la manière dont les propriétés de message sont définies. La structure est un paramètre d'entrée dans l'appel MQSETMP. Pour plus d'informations, voir [MQSMPO](#) .

### **Nom**

Type: MQCHARV-entrée

Il s'agit du nom de la propriété à définir.

Pour plus d'informations sur l'utilisation des noms de propriété, voir [Noms de propriété et Restrictions relatives aux noms de propriété](#) .

### **PropDesc**

Type: MQPD-entrée/sortie

Cette structure est utilisée pour définir les attributs d'une propriété, notamment:

- que se passe-t-il si la propriété n'est pas prise en charge
- quel est le contexte de message auquel appartient la propriété
- les messages dans lequel la propriété est copiée au fur et à mesure qu'elle est transmise

Pour plus d'informations sur cette structure, voir [MQPD](#) .

### **Tapez**

Type : MQLONG - entrée

Type de données de la propriété en cours de définition. Il peut s'agir de l'un des éléments suivants :

#### **MQTYPE\_BOOLÉEN**

Une valeur booléenne. *ValueLength* doit être 4.

#### **MQTYPE\_BYTE\_STRING**

Chaîne d'octets. *ValueLength* doit être supérieur ou égal à zéro.

#### **MQTYPE\_INT8**

Entier signé de 8 bits. *ValueLength* doit être 1.

#### **MQTYPE\_INT16**

Entier signé 16 bits. *ValueLength* doit être 2.

#### **MQTYPE\_INT32**

Entier signé 32 bits. *ValueLength* doit être 4.

#### **MQTYPE\_INT64**

Entier signé 64 bits. *ValueLength* doit être 8.

#### **MQTYPE\_FLOAT32**

Nombre à virgule flottante 32 bits. *ValueLength* doit être 4.

Remarque: ce type n'est pas pris en charge avec les applications utilisant IBM COBOL for z/OS.

#### **MQTYPE\_FLOAT64**

Nombre en virgule flottante 64 bits. *ValueLength* doit être 8.

Remarque: ce type n'est pas pris en charge avec les applications utilisant IBM COBOL for z/OS.

### **MQTYPE\_CHAINE**

Chaîne de caractères. *ValueLength* doit être supérieur ou égal à zéro ou la valeur spéciale MQVL\_NULL\_TERMINATED.

### **MQTYPE\_NULL**

La propriété existe mais a une valeur nulle. *ValueLength* doit être égal à zéro.

### **ValueLength**

Type : MQLONG - entrée

Longueur en octets de la valeur de propriété dans le paramètre *Valeur* . La valeur zéro est valide uniquement pour les valeurs null ou pour les chaînes ou les chaînes d'octets. La valeur zéro indique que la propriété existe mais que la valeur ne contient ni caractères ni octets.

La valeur doit être supérieure ou égale à zéro ou à la valeur spéciale suivante si MQTYPE\_STRING est défini pour le paramètre *Type* :

### **MQVL\_NULL\_TERMINATED**

La valeur est délimitée par la première valeur nulle rencontrée dans la chaîne. La valeur null n'est pas incluse dans la chaîne. Cette valeur n'est pas valide si MQTYPE\_STRING n'est pas également défini.

Remarque: Le caractère null utilisé pour terminer une chaîne si MQVL\_NULL\_TERMINATED est défini est une valeur null du jeu de caractères de la valeur.

### **Valeur**

Type: MQBYTEExValueLength-input

Valeur de la propriété à définir. La mémoire tampon doit être alignée sur une limite appropriée à la nature des données de la valeur.

Dans le langage de programmation C, le paramètre est déclaré comme un pointeur vers vide ; l'adresse de tout type de données peut être spécifiée comme paramètre.

Si *ValueLength* est égal à zéro, *Valeur* n'est pas référencé. Dans ce cas, l'adresse de paramètre transmise par les programmes écrits en assembleur C ou System/390 peut être null.

### **CompCode**

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

### **Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_WARNING:

#### **MQRC\_RFH\_FORMAT\_ERREUR**

(2421, X'0975') Un dossier MQRFH2 contenant des propriétés n'a pas pu être analysé.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'089C') Carte non disponible.

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

**MQRC\_ASID\_MISMATCH**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

**MQRC\_BUFFER\_ERROR**

(2004, X'07D4') Paramètre de valeur non valide.

**MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'07D5') Paramètre de longueur de valeur non valide.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

**ERREUR MQRC\_HMSG\_ERROR**

(2460, X'099C') Le pointeur de descripteur de message n'est pas valide.

**MQRC\_MSG\_HANDLE\_IN\_USE**

(2499, X'09C3') Descripteur de message déjà utilisé.

**MQRC\_OPTIONS\_ERROR**

(2046, X'07FE') Options non valides ou non cohérentes.

**MQRC\_PD\_ERREUR**

(2482, X'09B2') Structure de descripteur de propriété non valide.

**MQRC\_PROPERTY\_NAME\_ERROR**

(2442, X'098A') Nom de propriété non valide.

**MQRC\_PROPERTY\_TYPE\_ERROR**

(2473, X'09A9') Type de données de propriété non valide.

**MQRC\_PROP\_NUMBER\_FORMAT\_ERROR**

(2472, X'09A8') Erreur de format numérique détectée dans les données de valeur.

**MQRC\_SMPO\_ERREUR**

(2463, X'099F') La structure des options de propriété de message n'est pas valide.

**MQRC\_SOURCE\_CCSID\_ERROR**

(2111, X'083F') Identificateur de jeu de caractères codés de nom de propriété non valide.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') Mémoire disponible insuffisante.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Appel C

```
MQSETMP (Hconn, Hmsg, &SetPropOpts, &Name, &PropDesc, Type,
ValueLength, &Value, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;          /* Connection handle */
MQHMSG   Hmsg;          /* Message handle */
MQSMPO   SetPropOpts;  /* Options that control the action of MQSETMP */
MQCHARV  Name;         /* Property name */
MQPD     PropDesc;     /* Property descriptor */
MQLONG   Type;        /* Property data type */
MQLONG   ValueLength; /* Length of property value in Value */
MQBYTE   Value[n];    /* Property value */
MQLONG   CompCode;    /* Completion code */
MQLONG   Reason;      /* Reason code qualifying CompCode */
```

## Appel COBOL

```
CALL 'MQSETMP' USING HCONN, HMSG, SETMSGOPTS, NAME, PROPDESC, TYPE,
VALUELENGTH, VALUE, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle
01 HCONN PIC S9(9) BINARY.
** Message handle
01 HMSG PIC S9(18) BINARY.
** Options that control the action of MQSETMP
01 SETMSGOPTS.
COPY CMQSMPOV.
** Property name
01 NAME
COPY CMQCHRVA.
** Property descriptor
01 PROPDESC.
COPY CMQPDA.
** Property data type
01 TYPE PIC S9(9) BINARY.
** Length of property value in VALUE
01 VALUELENGTH PIC S9(9) BINARY.
** Property value
01 VALUE PIC X(n).
** Completion code
01 COMPCODE PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON PIC S9(9) BINARY.
```

## Appel PL/I

```
call MQSETMP (Hconn, Hmsg, SetPropOpts, Name, PropDesc, Type, ValueLength,
Value, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
dcl Hconn fixed bin(31); /* Connection handle */
dcl Hmsg fixed bin(63); /* Message handle */
dcl SetPropOpts like MQSMPO; /* Options that control the action of MQSETMP */
dcl Name like MQCHARV; /* Property name */
dcl PropDesc like MQPD; /* Property descriptor */
dcl Type fixed bin(31); /* Property data type */
dcl ValueLength fixed bin(31); /* Length of property value in Value */
dcl Value char(n); /* Property value */
dcl CompCode fixed bin(31); /* Completion code */
dcl Reason fixed bin(31); /* Reason code qualifying CompCode */
```

## Appel d'assembleur de haut niveau

```
CALL MQSETMP,(HCONN,HMSG,SETMSGHOPTS,NAME,PROPDESC,TYPE,VALUELENGTH,
VALUE,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

HCONN	DS	F	Connection handle
HMSG	DS	D	Message handle
SETMSGOPTS	CMQSMPOA	,	Options that control the action of MQSETMP
NAME	CMQCHRVA	,	Property name
PROPDESC	CMQPDA	,	Property descriptor
TYPE	DS	F	Property data type
VALUELENGTH	DS	F	Length of property value in VALUE
VALUE	DS	CL(n)	Property value

COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## MQSTAT-Extraction des informations de statut

Utilisez l'appel MQSTAT pour extraire les informations de statut. Le type des informations de statut renvoyées est déterminé par la valeur Type indiquée dans l'appel.

### Syntaxe

MQSTAT (*Hconn*, *Type*, *Stat*, *Code achèvement*, *Raison*)

### Paramètres

#### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS , l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour *Hconn*:

#### MQHC\_DEF\_HCONN

Descripteur de connexion par défaut.

#### Tapez

Type : MQLONG - entrée

Type d'informations de statut demandées. Les valeurs valides sont les suivantes:

#### MQSTAT\_TYPE\_ASYNC\_ERROR

Renvoie des informations sur les opérations d'insertion asynchrone précédentes.

#### MQSTAT\_TYPE\_RECONNEXION

Renvoie des informations sur la reconnexion. Si la connexion se reconnecte ou ne parvient pas à se reconnecter, les informations décrivent l'échec qui a provoqué la reconnexion de la connexion.

Cette valeur n'est valide que pour les connexions client. Pour les autres types de connexion, l'appel échoue avec le code anomalie **MQRC\_ENVIRONMENT\_ERROR**

#### MQSTAT\_TYPE\_RECONNECTION\_ERROR

Renvoie des informations sur un incident précédent lié à la reconnexion. Si la reconnexion a échoué, les informations décrivent l'échec qui a provoqué l'échec de la reconnexion.

Cette valeur n'est valide que pour les connexions client. Pour les autres types de connexion, l'appel échoue avec le code anomalie **MQRC\_ENVIRONMENT\_ERROR**.

#### Statistique

Type: MQSTS-entrée/sortie

Structure des informations de statut. Pour plus d'informations, voir [«MQSTS-Structure de génération de rapports de statut»](#), à la page 619.

#### CompCode

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

#### MQCC\_OK

Achèvement réussi.

#### MQCC\_FAILED

Echec de l'appel.

## Motif

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

### **MQRC\_API\_EXIT\_ERROR**

(2374, X' 946') Echec de l'exit API

### **MQRC\_API\_EXIT\_LOAD\_ERROR**

(2183, X'887') Impossible de charger l'exit API.

### **MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

### **MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

### **MQRC\_CONNECTION\_STOPPING**

(2203, X'89B') Connexion en cours d'arrêt.

### **MQRC\_FUNCTION\_NOT\_SUPPORTED**

(2298, X'8FA') La fonction demandée n'est pas disponible dans l'environnement en cours.

### **MQRC\_HCONN\_ERROR**

(2018, X'7E2') Descripteur de connexion non valide.

### **MQRC\_Q\_MGR\_STOPPING**

(2162, X'872')-Arrêt du gestionnaire de files d'attente

### **MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') Ressources système disponibles insuffisantes.

### **MQRC\_STAT\_TYPE\_ERREUR**

(2430, X'97E') Erreur avec le type MQSTAT

### **MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') Mémoire disponible insuffisante.

### **MQRC\_STS\_ERREUR**

(2426, X'97A') Erreur avec la structure MQSTS

### **MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Remarques sur l'utilisation

1. Un appel à MQSTAT spécifiant un type de MQSTAT\_TYPE\_ASYNC\_ERROR renvoie des informations sur les opérations précédentes asynchrones MQPUT et MQPUT1 . La structure MQSTS retransmise lors du retour de l'appel MQSTAT contient les premières informations d'avertissement ou d'erreur asynchrone enregistrées pour cette connexion. Si d'autres erreurs ou avertissements suivent le premier, ils ne modifient normalement pas ces valeurs. Toutefois, si une erreur se produit avec le code achèvement MQCC\_WARNING, un échec ultérieur avec le code achèvement MQCC\_FAILED est renvoyé à la place.
2. Si aucune erreur ne s'est produite depuis l'établissement de la connexion ou depuis le dernier appel à MQSTAT , un CompCode de MQCC\_OK et une raison de MQRC\_NONE sont renvoyés dans la structure MQSTS .
3. Le nombre d'appels asynchrones traités sous le descripteur de connexion est renvoyé par l'intermédiaire de trois zones de compteur: PutSuccessCount, PutWarningCount et PutFailureCount. Ces compteurs sont incrémentés par le gestionnaire de files d'attente chaque fois qu'une opération asynchrone est traitée avec succès, a un avertissement ou échoue (notez qu'à

des fins de comptabilité, une insertion dans une liste de distribution compte une fois par file d'attente de destination plutôt qu'une fois par liste de distribution). Un compteur n'est pas incrémenté au-delà de la valeur positive maximale AMQ\_LONG\_MAX.

4. Un appel réussi à MQSTAT entraîne la réinitialisation des informations d'erreur ou des comptages précédents.
5. Le comportement de MQSTAT dépend de la valeur du paramètre **MQSTAT Type** que vous indiquez.

#### 6. **MQSTAT\_TYPE\_ASYNC\_ERROR**

- a. Un appel à MQSTAT spécifiant un type de MQSTAT\_TYPE\_ASYNC\_ERROR renvoie des informations sur les opérations précédentes asynchrones MQPUT et MQPUT1 . La structure MQSTS retransmise lors du retour de l'appel MQSTAT contient les premières informations d'avertissement ou d'erreur asynchrone enregistrées pour cette connexion. Si d'autres erreurs ou avertissements suivent le premier, ils ne modifient normalement pas ces valeurs. Toutefois, si une erreur se produit avec le code achèvement MQCC\_WARNING, un échec ultérieur avec le code achèvement MQCC\_FAILED est renvoyé à la place.
- b. Si aucune erreur ne s'est produite depuis l'établissement de la connexion ou depuis le dernier appel à MQSTAT , un CompCode de MQCC\_OK et une raison de MQRC\_NONE sont renvoyés dans la structure MQSTS .
- c. Le nombre d'appels asynchrones traités sous le descripteur de connexion est renvoyé par l'intermédiaire de trois zones de compteur: PutSuccessCount, PutWarningCount et PutFailureCount. Ces compteurs sont incrémentés par le gestionnaire de files d'attente chaque fois qu'une opération asynchrone est traitée avec succès, a un avertissement ou échoue (notez qu'à des fins de comptabilité, une insertion dans une liste de distribution compte une fois par file d'attente de destination plutôt qu'une fois par liste de distribution). Un compteur n'est pas incrémenté au-delà de la valeur positive maximale AMQ\_LONG\_MAX.
- d. Un appel réussi à MQSTAT entraîne la réinitialisation des informations d'erreur ou des comptages précédents.

#### **MQSTAT\_TYPE\_RECONNEXION**

Supposons que vous appelez MQSTAT avec Type défini sur MQSTAT\_TYPE\_RECONNECTION dans un gestionnaire d'événements lors de la reconnexion. Tenez compte de ces exemples.

##### **Le client tente de se reconnecter ou n'a pas réussi à se reconnecter.**

CompCode dans la structure MQSTS est MQCC\_FAILED et Reason peut être MQRC\_CONNECTION\_BROKEN ou MQRC\_Q\_MGR QUIESCING. ObjectType est MQOT\_Q\_MGR, ObjectName est le nom du gestionnaire de files d'attente et ObjectQMgrName est vide.

##### **La reconnexion du client a abouti ou n'a jamais été déconnectée.**

CompCode dans la structure MQSTS est MQCC\_OK et Reason est MQRC\_NONE

Les appels ultérieurs à MQSTAT renvoient les mêmes résultats.

#### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

Supposons que vous appelez MQSTAT avec Type défini sur MQSTAT\_TYPE\_RECONNECTION\_ERROR en réponse à la réception de MQRC\_RECONNECT\_FAILED dans un appel MQI. Tenez compte de ces exemples.

##### **Une erreur d'autorisation s'est produite lors de la réouverture d'une file d'attente lors de la reconnexion à un autre gestionnaire de files d'attente.**

CompCode dans la structure MQSTS est MQCC\_FAILED et Reason est la raison pour laquelle la reconnexion a échoué, par exemple MQRC\_NOT\_AUTHORIZED. ObjectType est le type d'objet à l'origine du problème, par exemple MQOT\_QUEUE, ObjectName est le nom de la file d'attente et ObjectQMgrName le nom du gestionnaire de files d'attente propriétaire de la file d'attente.

##### **Une erreur de connexion socket s'est produite lors de la reconnexion.**

CompCode dans la structure MQSTS est MQCC\_FAILED et Reason est la raison pour laquelle la reconnexion a échoué, par exemple MQRC\_HOST\_NOT\_AVAILABLE. ObjectType est MQOT\_Q\_MGR, ObjectName est le nom du gestionnaire de files d'attente et ObjectQMgrName est vide.

Les appels ultérieurs à MQSTAT renvoient les mêmes résultats.

## Appel C

```
MQSTAT (Hconn, StatType, &Stat, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN Hconn;          /* Connection Handle */
MQLONG StatType;       /* Status type */
MQSTS Stat;           /* Status information structure */
MQLONG CompCode;      /* Completion code */
MQLONG Reason;        /* Reason code qualifying CompCode */
```

## Appel COBOL

```
CALL 'MQSTAT' USING HCONN, STATTYPE, STAT, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**      Connection handle
01     HCONN      PIC S9(9)      BINARY.
**      Status type
01     STATTYPE  PIC S9(9)      BINARY.
**      Status information
01     STAT.
      COPY CMQSTSV.
**      Completion code
01     COMPCODE  PIC S9(9)      BINARY.
**      Reason code qualifying COMPCODE
01     REASON    PIC S9(9)      BINARY.
```

## Appel PL/I

```
call MQSTAT (Hconn, StatType, Stat, Compcode, Reason);
```

Déclarez les paramètres comme suit :

```
dcl Hconn      fixed bin(31); /* Connection handle */
dcl StatType   fixed bin(31); /* Status type */
dcl Stat       like MQSTS;    /* Status information structure */
dcl CompCode   fixed bin(31); /* Completion code */
dcl Reason     fixed bin(31); /* Reason code qualifying CompCode */
```

## Appel de l'assembleur System/390

```
CALL MQSTAT,(HCONN,STATTYPE,STAT,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
HCONN      DS      F      Connection handle
STATTYPE   DS      F      Status type
STAT       CMQSTSA,  Status information structure
COMPCODE   DS      F      Completion code
REASON     DS      F      Reason code qualifying COMPCODE
```



## MQSUB-Enregistrement d'abonnement

Utilisez l'appel MQSUB pour enregistrer l'abonnement des applications à une rubrique particulière.

### Syntaxe

MQSUB (*Hconn*, *SubDesc*, *Hobj*, *Hsub*, *Code achèvement*, *Motif*)

### Paramètres

#### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS, l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour *Hconn*:

#### MQHC\_DEF\_HCONN

Descripteur de connexion par défaut.

#### SubDesc

Type: MQSD-entrée/sortie

Il s'agit d'une structure qui identifie l'objet en cours d'utilisation qui est enregistré par l'application. Pour plus d'informations, voir «MQSD - Descripteur d'abonnement», à la page 594.

#### Hobj

Type: MQHOBJ-entrée / sortie

Ce descripteur représente l'accès qui a été établi pour obtenir les messages envoyés à cet abonnement. Ces messages peuvent être stockés dans une file d'attente spécifique ou le gestionnaire de files d'attente peut gérer leur stockage sans utiliser de file d'attente spécifique.

Pour utiliser une file d'attente spécifique, vous devez l'associer à l'abonnement lors de sa création. Vous pouvez effectuer cette opération de deux manières :

- En utilisant la commande DEFINE SUB MQSC et en indiquant cette commande avec le nom d'un objet file d'attente.
- En fournissant ce descripteur lors de l'appel de MQSUB avec MQSO\_CREATE

Si ce descripteur est fourni en tant que paramètre d'entrée dans l'appel, il doit s'agir d'un descripteur d'objet valide renvoyé par un appel MQOPEN précédent d'une file d'attente à l'aide d'au moins l'une des options suivantes:

- MQOO\_ENTRÉE\_\*
- MQOO\_BROWSE
- MQOO\_OUTPUT (si la file d'attente est une file d'attente éloignée)

Si tel n'est pas le cas, l'appel échoue avec MQRC\_HOBJ\_ERROR. Il ne peut pas s'agir d'un descripteur d'objet dans une file d'attente alias qui se résout en un objet de rubrique. Si tel est le cas, l'appel échoue avec MQRC\_HOBJ\_ERROR.

Si le gestionnaire de files d'attente doit gérer le stockage des messages envoyés à cet abonnement, il doit être défini lors de la création de l'abonnement, à l'aide de l'option MQSO\_MANAGED. Le gestionnaire de files d'attente renvoie ensuite cet identificateur en tant que paramètre de sortie sur l'appel. Le descripteur renvoyé est appelé descripteur géré. Si MQHO\_NONE est spécifié mais que MQSO\_MANAGED n'est pas spécifié, l'appel échoue avec MQRC\_HOBJ\_ERROR.

Lorsqu'un descripteur géré vous est renvoyé par le gestionnaire de files d'attente, vous pouvez l'utiliser sur un appel MQGET ou MQCB avec ou sans options de navigation, sur un appel MQINQ ou sur MQCLOSE. Vous ne pouvez pas l'utiliser sur MQPUT, MQSUB, MQSET ; la tentative échoue avec MQRC\_NOT\_OPEN\_FOR\_OUTPUT, MQRC\_HOBJ\_ERROR ou MQRC\_NOT\_OPEN\_FOR\_SET.

Si cet abonnement est repris à l'aide de l'option MQSO\_RESUME dans la structure MQSD, le descripteur peut être renvoyé à l'application dans ce paramètre en définissant MQSO\_MANAGED sur MQHO\_NONE. Vous pouvez le faire, que l'abonnement utilise un descripteur géré ou non, et il peut être utile de fournir des abonnements créés à l'aide de DEFINE SUB avec le descripteur de la file d'attente d'abonnement définie sur cette commande. Dans le cas où un abonnement créé par l'administrateur est repris, la file d'attente s'ouvre avec MQOO\_INPUT\_AS\_Q\_DEF et MQOO\_BROWSE. Si vous devez spécifier d'autres options, l'application doit ouvrir la file d'attente d'abonnement de manière explicite et fournir le descripteur d'objet lors de l'appel. En cas de problème lors de l'ouverture de la file d'attente, l'appel échoue avec MQRC\_INVALID\_DESTINATION. Si *Hobj* est fourni, il doit être équivalent à *Hobj* dans l'appel MQSUB d'origine. Cela signifie que si un descripteur d'objet renvoyé par un appel MQOPEN est fourni, il doit être placé dans la même file d'attente que celle utilisée précédemment. S'il ne s'agit pas de la même file d'attente, l'appel échoue avec MQRC\_HOBJ\_ERROR.

Si cet abonnement est modifié à l'aide de l'option MQSO\_ALTER dans la structure MQSD, un *Hobj* différent peut être fourni. Toutes les publications qui ont été distribuées à la file d'attente et qui ont été précédemment identifiées à l'aide de ce paramètre restent dans cette file d'attente et il incombe à l'application d'extraire ces messages si le paramètre **Hobj** représente désormais une file d'attente différente.

Tableau 555. Utilisation de <i>hobj</i> avec différentes options d'abonnement		
Options	<i>Hobj</i>	Description
MQSO_CREATE + MQSO_MANAGED	Ignoré en entrée	Crée un abonnement avec le stockage des messages gérés par le gestionnaire de files d'attente
MQSO_CREER	Un descripteur d'objet valide	Crée un abonnement fournissant une file d'attente spécifique comme destination des messages.
MQSO_RESUME	MQHO_AUCUN	Reprend un abonnement précédemment créé, qu'il ait été géré ou non, et demande au gestionnaire de files d'attente de renvoyer le descripteur d'objet à utiliser par l'application.
MQSO_RESUME	Descripteur d'objet de correspondance valide	Reprend un abonnement créé précédemment qui utilise une file d'attente spécifique comme destination des messages et utilise un descripteur d'objet avec des options d'ouverture spécifiques.
MQSO_ALTER + MQSO_MANAGED	MQHO_AUCUN	Modifie un abonnement existant qui utilisait auparavant une file d'attente spécifique. Il s'agit donc désormais d'un abonnement géré. La classe de destination (gérée ou non) ne peut pas être modifiée.

Tableau 555. Utilisation de *hobj* avec différentes options d'abonnement (suite)

Options	Hobj	Description
MQSO_ALTER	Un descripteur d'objet valide	Modifie un abonnement existant, qu'il ait été géré ou non, de sorte qu'il utilise désormais une file d'attente spécifique. Lorsque l'option MQSO_MANAGED n'est pas utilisée, la file d'attente fournie peut être modifiée, mais la classe de destination (gérée ou non) ne peut pas être modifiée.

Qu'il ait été fourni ou renvoyé, *Hobj* doit être spécifié lors des appels MQGET ou MQCB suivants qui souhaitent recevoir les messages de publication envoyés à cet abonnement.

Le descripteur *Hobj* n'est plus valide lorsque l'appel MQCLOSE est émis dessus ou lorsque l'unité de traitement qui définit la portée du descripteur s'arrête (jusqu'à ce que l'application se déconnecte). La portée du descripteur d'objet renvoyé est identique à celle du descripteur de connexion spécifié dans l'appel. Pour plus d'informations sur la portée de descripteur, voir *Hconn (MQHCONN)-output*. Une exception MQCLOSE de l'identificateur *Hobj* n'affecte pas l'identificateur *Hsub*.

### HSub

Type: MQHOBJ-sortie

Ce descripteur représente l'abonnement qui a été effectué. Il peut être utilisé pour deux autres opérations:

- Il peut être utilisé lors d'un appel MQSUBRQ ultérieur pour demander l'envoi de publications lorsque l'option MQSO\_PUBLICATIONS\_ON\_REQUEST a été utilisée lors de la création de l'abonnement.
- Il peut être utilisé lors d'un appel MQCLOSE ultérieur pour supprimer l'abonnement qui a été effectué. Le descripteur *Hsub* cesse d'être valide lorsque l'appel MQCLOSE est émis ou lorsque l'unité de traitement qui définit la portée du descripteur s'arrête. La portée du descripteur d'objet renvoyé est identique à celle du descripteur de connexion spécifié dans l'appel. Une exception MQCLOSE de l'identificateur *Hsub* n'affecte pas l'identificateur *Hobj*.

Ce descripteur ne peut pas être transmis à un appel MQGET ou MQCB. Vous devez utiliser le paramètre **Hobj**. Vous ne pouvez pas utiliser ce descripteur sur un appel IBM MQ autre que MQCLOSE ou MQSUBRQ. La transmission de ce descripteur à tout autre appel IBM MQ génère une erreur MQRC\_HOBJ\_ERROR.

### CompCode

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

#### MQCC\_OK

Achèvement réussi

#### MQCC\_WARNING

Avertissement (achèvement partiel)

#### MQCC\_FAILED

Echec de l'appel

### Motif

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK, le code anomalie est le suivant:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED, le code anomalie est l'un des suivants:

**Erreur de résolution MQRC\_CLUSTER\_RESOLUTION\_ERROR**

(2189, X'88D') La résolution du nom de cluster a échoué.

**MQRC\_DURABILITY\_NOT\_ALLOWED**

2436 (X'0984') Un appel MQSUB utilisant l'option MQSO\_DURABLE a échoué.

**MQRC\_FUNCTION\_NOT\_SUPPORTED**

2298 (X'08FA') La fonction demandée n'est pas disponible dans l'environnement en cours.

**MQRC\_HOBJ\_ERROR**

2019 (X'07E3') Descripteur d'objet Hobj non valide.

**MQRC\_IDENTITY\_MISMATCH**

2434 (X'0982') Le nom de l'abonnement correspond à l'abonnement existant.

**MQRC\_NOT\_AUTHORIZED**

2035 (X'07F3') L'utilisateur n'est pas autorisé à effectuer l'opération.

**MQRC\_NO\_ABONNEMENT**

2428 (X'097C') Le nom d'abonnement identifié n'existe pas.

**MQRC\_OBJECT\_STRING\_ERROR**

2441 (X'0989') Zone Objectstring non valide.

**MQRC\_OPTIONS\_ERROR**

2046 (X'07FE') Le paramètre ou la zone Options contient des options non valides ou une combinaison d'options non valides.

**MQRC\_Q\_MGR QUIESCING**

2161 (X'0871') Mise au repos du gestionnaire de files d'attente.

**MQRC\_RECONNECT\_Q\_MGR\_REQD**

2555 (X'09FB'X) L'option MQCNO\_RECONNECT\_Q\_MGR est requise.

**MQRC\_RETAINED\_MSG\_Q\_ERREUR**

2525 (X'09DD') Les publications conservées qui existent pour la chaîne de rubrique souscrite ne peuvent pas être extraites.

**MQRC\_RETAINED\_NOT LIVRÉ**

2526 (X'09DE') Les publications conservées qui existent pour la chaîne de rubrique souscrite ne peuvent pas être distribuées à la file d'attente de destination de l'abonnement et ne peuvent pas être distribuées à la file d'attente de rebut.

**MQRC\_SD\_ERREUR**

2424 (X'0978') Descripteur d'abonnement (MQSD) non valide.

**MQRC\_SELECTION\_NOT\_AVAILABLE**

2551 (X'09F7') La chaîne de sélection ne respecte pas la syntaxe du sélecteur IBM MQ et aucun fournisseur de sélection de messages étendue n'est disponible.

**MQRC\_SELECTION\_STRING\_ERROR**

2519 (X'09D7') La chaîne de sélection doit être spécifiée comme décrit dans la documentation de la structure MQCHARV.

**MQRC\_SELECTOR\_SYNTAX\_ERREUR**

2459 (X'099B') Un appel MQOPEN, MQPUT1 ou MQSUB a été émis, mais une chaîne de sélection contenant une erreur de syntaxe a été spécifiée.

**MQRC\_SUB\_USER\_DATA\_ERROR**

2431 (X'097F') Zone de données SubUser non valide.

**MQRC\_SUB\_NAME\_ERREUR**

2440 (X'0988') Zone SubName non valide.

**MQRC\_SUB\_ALREADY\_EXISTS**

2432 (X'0980') L'abonnement existe déjà.

**MQRC\_SUB\_USER\_DATA\_ERROR**

2431 (X'097F') Zone de données SubUser non valide.

**MQRC\_TOPIC\_STRING\_ERROR**

2425 (X'0979') La chaîne de rubrique n'est pas valide.

**MQRC\_UNKNOWN\_NOM\_OBJET**

2085 (X'0825') L'objet identifié dans la zone ObjectName de MQSD est introuvable.

**MQRC\_SUB\_JOIN\_NOT\_ALTERABLE**

29440 (X'7300') Le mode de partage d'abonnement est incompatible avec l'abonnement existant. Cette erreur peut être renvoyée lors de la tentative de reprise d'un abonnement partagé JMS 2.0 dans une application non JMS.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Remarques sur l'utilisation

- L'abonnement est effectué à une rubrique, nommée à l'aide du nom abrégé d'un objet de rubrique prédéfini, du nom complet de la chaîne de rubrique ou de la concaténation de deux parties. Voir la description de *ObjectName* et *ObjectString* dans «MQSD - Descripteur d'abonnement», à la page 594.
- Le gestionnaire de files d'attente effectue des contrôles de sécurité lorsqu'un appel MQSUB est émis pour vérifier que l'ID utilisateur sous lequel l'application s'exécute dispose du niveau de droits approprié avant que l'accès ne soit autorisé. L'objet de rubrique approprié se trouve dans la hiérarchie de rubriques et une vérification des droits d'accès est effectuée sur cet objet de rubrique pour s'assurer que les droits d'abonnement sont définis. Si l'option MQSO\_MANAGED n'est pas utilisée, une vérification des droits d'accès est effectuée sur la file d'attente de destination pour s'assurer que les droits d'accès à la sortie sont définis. Si l'option MQSO\_MANAGED est utilisée, aucune vérification des droits d'accès n'est effectuée sur la file d'attente gérée pour la sortie ou l'accès en interrogation.
- Si vous ne fournissez pas d'entrée Hobj, l'appel MQSUB alloue deux descripteurs, un descripteur d'objet (Hobj) et un descripteur d'abonnement (Hsub).
- Le Hobj renvoyé dans l'appel MQSUB lorsque l'option MQSO\_MANAGED est utilisée, peut être interrogé afin de trouver des attributs tels que le seuil d'annulation et le nom de remise en file d'attente d'annulation excessive. Vous pouvez également demander le nom de la file d'attente gérée, mais vous ne devez pas tenter d'ouvrir directement cette file d'attente.
- Les abonnements peuvent être regroupés, ce qui permet de distribuer une seule publication au groupe d'abonnements même si plusieurs groupes correspondent à la publication. Les abonnements sont regroupés à l'aide de l'option MQSO\_GROUP\_SUB et, pour pouvoir regrouper les abonnements, ils doivent être
  - utilisation de la même file d'attente nommée (qui n'utilise pas l'option MQSO\_MANAGED) sur le même gestionnaire de files d'attente-représenté par le paramètre Hobj de l'appel MQSUB
  - partager le même ID SubCorrel
  - être du même SubLevelCes attributs définissent l'ensemble des abonnements considérés comme faisant partie du groupe et sont également les attributs qui ne peuvent pas être modifiés si un abonnement est groupé. L'altération de SubLevel entraîne MQRC\_SUBLEVEL\_NOT\_ALTERABLE, et l'altération de l'un des autres (qui peut être modifié si un abonnement n'est pas groupé) entraîne MQRC\_GROUPING\_NOT\_ALTERABLE.
- La réussite de l'appel MQSUB ne signifie pas que l'action a abouti. Pour vérifier que cet appel est terminé, voir l'étape [DEFINE SUB](#) dans [Vérification de la fin des commandes asynchrones pour les réseaux distribués](#).
- Les zones du MQSD sont renseignées en cas de retour d'un appel MQSUB qui utilise l'option MQSO\_RESUME. Le MQSD renvoyé peut être transmis directement dans un appel MQSUB qui utilise l'option MQSO\_ALTER avec les modifications que vous devez apporter à l'abonnement appliqué au MQSD. Certaines zones comportent des considérations spéciales, comme indiqué dans le tableau.

Tableau 556. Considérations spéciales pour les zones du MQSD

Nom de zone dans MQSD	Des considérations spéciales
Options d'accès ou de création	Certaines options peuvent être réinitialisées en cas de retour à partir de l'appel MQSUB. Si vous réutilisez ensuite le MQSD dans un appel MQSUB, l'option requise doit être explicitement définie.
Options de durabilité, Options de destination, Options d'enregistrement et Options de caractère générique	Ces options sont définies comme il convient
Options de publication	Ces options sont définies comme il convient, à l'exception de MQSO_NEW_PUBLICATIONS_ONLY, qui s'applique uniquement à MQSO_CREATE.
Autres options	Ces options sont inchangées en cas de retour d'un appel MQSUB. Ils contrôlent la façon dont l'appel d'API est émis et ne sont pas stockés avec l'abonnement. Ils doivent être définis comme requis lors de tout appel MQSUB ultérieur réutilisant le MQSD.
ObjectName	Cette zone d'entrée uniquement n'est pas modifiée en cas de retour d'un appel MQSUB.
ObjectString	Cette zone d'entrée uniquement n'est pas modifiée en cas de retour d'un appel MQSUB. Le nom de rubrique complet utilisé est renvoyé dans la zone <i>ResObjectString</i> , si une mémoire tampon est fournie.
ID AlternateUser et ID AlternateSecurity	Ces zones d'entrée uniquement sont inchangées en cas de retour d'un appel MQSUB. Ils contrôlent la façon dont l'appel d'API est émis et ne sont pas stockés avec l'abonnement. Ils doivent être définis comme requis lors de tout appel MQSUB ultérieur réutilisant le MQSD.
SubExpiry	En cas de retour d'un appel MQSUB à l'aide de l'option MQSO_RESUME, cette zone est définie sur l'expiration d'origine de l'abonnement et non sur l'heure d'expiration restante. Si vous réutilisez ensuite le MQSD dans un appel MQSUB à l'aide de l'option MQSO_ALTER, vous réinitialisez l'expiration de l'abonnement pour recommencer le décomptage.
SubName	Cette zone est une zone d'entrée dans un appel MQSUB et n'est pas modifiée dans la sortie.

Tableau 556. Considérations spéciales pour les zones du MQSD (suite)

Nom de zone dans MQSD	Des considérations spéciales
SubUserData et SelectionString	<p>Ces zones de longueur variable sont renvoyées dans la sortie d'un appel MQSUB à l'aide de l'option MQSO_RESUME, si une mémoire tampon est fournie, ainsi qu'une longueur de mémoire tampon positive dans <i>VSubfSize</i>. Si aucune mémoire tampon n'est fournie, seule la longueur est renvoyée dans la zone <i>VSLength</i> de MQCHARV. Si la mémoire tampon fournie est inférieure à l'espace requis pour renvoyer la zone, seuls <i>VSubfSize</i> octets sont renvoyés dans la mémoire tampon fournie.</p> <p>Si vous réutilisez ensuite le MQSD dans un appel MQSUB à l'aide de l'option MQSO_ALTER et qu'une mémoire tampon n'est pas fournie, mais qu'une valeur <i>VSLength</i> différente de zéro est fournie, si cette longueur correspond à la longueur existante de la zone, aucune modification n'est apportée à la zone.</p>
ID SubCorrelet jeton PubAccounting	<p>Si vous n'utilisez pas MQSO_SET_CORREL_ID, <i>SubCorrelId</i> est généré par le gestionnaire de files d'attente. Si vous n'utilisez pas MQSO_SET_IDENTITY_CONTEXT, le <i>PubAccountingToken</i> est généré par le gestionnaire de files d'attente.</p> <p>Ces zones sont renvoyées dans le MQSD à partir d'un appel MQSUB à l'aide de l'option MQSO_RESUME. Si elles sont générées par le gestionnaire de files d'attente, la valeur générée est renvoyée dans un appel MQSUB à l'aide de l'option MQSO_CREATE ou MQSO_ALTER.</p>
PubPriority, SubLevel & PubApplIdentityData	Ces zones sont renvoyées dans le MQSD.
Chaîne ResObject	Cette zone de sortie uniquement est renvoyée dans le MQSD si une mémoire tampon est fournie.

## Appel C

```
MQSUB (Hconn, &SubDesc, &Hobj, &Hsub, &CompCode, &Reason)
```

Déclarez les paramètres comme suit :

```
MQHCONN Hconn; /* Connection handle */
MQSD SubDesc; /* Subscription descriptor */
MQHOBJ Hobj; /* Object handle */
MQHOBJ Hsub; /* Subscription handle */
MQLONG CompCode; /* Completion code */
MQLONG Reason; /* Reason code qualifying CompCode */
```

## Appel COBOL

```
CALL 'MQSUB' USING HCONN, SUBDESC, HOBJ, HSUB, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle
01 HCONN    PIC S9(9) BINARY.
** Subscription descriptor
01 SUBDESC.
   COPY CMQSDV.
** Object handle
01 HOBJ     PIC S9(9) BINARY.
** Subscription handle
01 HSUB     PIC S9(9) BINARY.
** Completion code
01 COMPCODE PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON   PIC S9(9) BINARY.
```

## Appel PL/I

```
call MQSUB (Hconn, SubDesc, Hobj, Hsub, CompCode, Reason)
```

Déclarez les paramètres comme suit :

```
dcl Hconn    fixed bin(31); /* Connection handle */
dcl SubDesc  like MQSD;    /* Subscription descriptor */
dcl Hobj     fixed bin(31); /* Object handle */
dcl Hsub     fixed bin(31); /* Subscription handle */
dcl CompCode fixed bin(31); /* Completion code */
dcl Reason   fixed bin(31); /* Reason code qualifying CompCode */
```

## Appel d'assembleur de haut niveau

```
CALL MQSUB, (HCONN, SUBDESC, HOBJ, HSUB, COMPCODE, REASON)
```

Déclarez les paramètres comme suit :

```
HCONN    DS      F  Connection handle
SUBDESC  CMQSDA  ,  Subscription descriptor
HOBJ     DS      F  Object handle
HSUB     DS      F  Subscription handle
COMPCODE DS      F  Completion code
REASON   DS      F  Reason code qualifying COMPCODE
```

## MQSUBRQ-Demande d'abonnement

Utilisez l'appel MQSUBRQ pour demander la publication conservée lorsque l'abonné a été enregistré auprès de MQSO\_PUBLICATIONS\_ON\_REQUEST.

### Syntaxe

MQSUBRQ (*Hconn*, *Hsub*, *Action*, *SubRqOpts*, *Code achèvement*, *Motif*)

### Paramètres

#### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS, l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour *Hconn*:



## **MQHC\_DEF\_HCONN**

Descripteur de connexion par défaut.

### **HSub**

Type : MQHOBJ - entrée

Ce descripteur représente l'abonnement pour lequel une mise à jour doit être demandée. La valeur de *Hsub* a été renvoyée à partir d'un appel MQSUB précédent.

### **Action**

Type : MQLONG - entrée

Ce paramètre contrôle l'action particulière demandée sur l'abonnement. La valeur suivante doit être spécifiée:

#### **PUBLICATION MQSR\_ACTION\_PUBLICATION**

Cette action demande qu'une publication de mise à jour soit envoyée pour la rubrique spécifiée. Il ne peut être utilisé que si l'abonné a spécifié l'option MQSO\_PUBLICATIONS\_ON\_REQUEST sur l'appel MQSUB lorsqu'il a effectué l'abonnement. Si le gestionnaire de files d'attente possède une publication conservée pour la rubrique, celle-ci est envoyée à l'abonné. Si ce n'est pas le cas, l'appel échoue. Si une application reçoit une publication qui a été conservée, elle est indiquée par la propriété de message MQIsRetained de cette publication.

Etant donné que la rubrique de l'abonnement existant représenté par le paramètre *Hsub* peut contenir des caractères génériques, l'abonné peut recevoir plusieurs publications conservées.

### **Options SubRq**

Type: MQSRO-entrée / sortie

Ces options contrôlent l'action de MQSUBRQ. Pour plus de détails, voir [«MQSRO-Options de demande d'abonnement»](#), à la page 616 .

Si aucune option n'est requise, les programmes écrits en assembleur C ou S/390 peuvent spécifier une adresse de paramètre null au lieu de spécifier l'adresse d'une structure MQSRO.

### **CompCode**

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

#### **MQCC\_OK**

Achèvement réussi

#### **MQCC\_WARNING**

Avertissement (achèvement partiel)

#### **MQCC\_FAILED**

Echec de l'appel

### **Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_FUNCTION\_NOT\_SUPPORTED**

2298 (X'08FA') La fonction demandée n'est pas disponible dans l'environnement en cours.

#### **MQRC\_NO\_RETAINED\_MSG**

2437 (X'0985') Aucune publication conservée n'est actuellement stockée pour cette rubrique.

**MQRC\_OPTIONS\_ERROR**

2046 (X'07FE') Le paramètre ou la zone Options contient des options non valides ou une combinaison d'options non valides.

**MQRC\_Q\_MGR QUIESCING**

2161 (X'0871') Mise au repos du gestionnaire de files d'attente.

**MQRC\_SRO\_ERREUR**

2438 (X'0986') Sur l'appel MQSUBRQ, les options de demande d'abonnement MQSRO ne sont pas valides.

**MQRC\_RETAINED\_MSG\_Q\_ERREUR**

2525 (X'09DD') Les publications conservées qui existent pour la chaîne de rubrique souscrite ne peuvent pas être extraites.

**MQRC\_RETAINED\_NOT LIVRÉ**

2526 (X'09DE') Les publications conservées qui existent pour la chaîne de rubrique souscrite ne peuvent pas être distribuées à la file d'attente de destination de l'abonnement et ne peuvent pas être distribuées à la file d'attente de rebut.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

**Remarques sur l'utilisation**

Les remarques de syntaxe suivantes s'appliquent à l'utilisation du code d'action MQSR\_ACTION\_PUBLICATION:

1. Si cette instruction aboutit, les publications conservées correspondant à l'abonnement spécifié ont été envoyées à l'abonnement et peuvent être reçues à l'aide de MQGET ou de MQCB à l'aide du Hobj renvoyé dans l'instruction MQSUB d'origine qui a créé l'abonnement.
2. Si la rubrique à laquelle est abonné l'instruction MQSUB d'origine qui a créé l'abonnement contient un caractère générique, plusieurs publications conservées peuvent être envoyées. Le nombre de publications envoyées à la suite de cet appel est enregistré dans la zone NumPubs de la structure SubRqOpts.
3. Si cette instruction se termine avec le code anomalie MQRC\_NO\_RETAINED\_MSG, cela signifie qu'il n'y a pas de publications actuellement conservées pour la rubrique spécifiée. #
4. Si cette instruction se termine avec le code anomalie MQRC\_RETAINED\_MSG\_Q\_ERROR ou MQRC\_RETAINED\_NOT LIVRÉES, des publications sont actuellement conservées pour la rubrique spécifiée, mais une erreur s'est produite indiquant qu'elles n'ont pas pu être distribuées.
5. L'application doit disposer d'un abonnement en cours à la rubrique pour pouvoir effectuer cet appel. Si l'abonnement a été effectué dans une instance précédente de l'application et qu'un descripteur valide de l'abonnement n'est pas disponible, l'application doit d'abord appeler MQSUB avec l'option MQSO\_RESUME afin d'obtenir un descripteur pour l'utiliser dans cet appel.
6. Les publications sont envoyées à la destination enregistrée pour être utilisée avec l'abonnement en cours de cette application. Si les publications doivent être envoyées ailleurs, l'abonnement doit d'abord être modifié à l'aide de l'appel MQSUB avec l'option MQSO\_ALTER.

**Appel C**

```
MQSUB (Hconn, Hsub, Action, &SubRqOpts, &CompCode, &Reason)
```

Déclarez les paramètres comme suit :

```
MQHCONN Hconn;      /* Connection handle */
MQHOBJ   Hsub;       /* Subscription handle */
MQLONG   Action;     /* Action requested by MQSUBRQ */
MQSRO    SubRqOpts; /* Options that control the action of MQSUBRQ */
MQLONG   CompCode;  /* Completion code */
MQLONG   Reason;    /* Reason code qualifying CompCode */
```

## Appel COBOL

```
CALL 'MQSUBRQ' USING HCONN, HSUB, ACTION, SUBRQOPTS, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle  
01 HCONN PIC S9(9) BINARY.  
** Subscription handle  
01 HSUB PIC S9(9) BINARY.  
** Action requested by MQSUBRQ  
01 ACTION PIC S9(9) BINARY.  
** Options that control the action of MQSUBRQ  
01 SUBRQOPTS.  
COPY CMQSROV.  
** Completion code  
01 COMPCODE PIC S9(9) BINARY.  
** Reason code qualifying COMPCODE  
01 REASON PIC S9(9) BINARY.
```

## Appel PL/I

```
call MQSUBRQ (Hconn, Hsub, Action, SubRqOpts, CompCode, Reason)
```

Déclarez les paramètres comme suit :

```
dcl Hconn fixed bin(31); /* Connection handle */  
dcl Hsub fixed bin(31); /* Subscription handle */  
dcl Action fixed bin(31); /* Action requested by MQSUBRQ */  
dcl SubRqOpts like MQR0; /* Options that control the action of MQSUBRQ */  
dcl CompCode fixed bin(31); /* Completion code */  
dcl Reason fixed bin(31); /* Reason code qualifying CompCode */
```

## Appel d'assembleur de haut niveau

```
CALL MQSUBRQ,(HCONN, HSUB, ACTION, SUBRQOPTS,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
HCONN DS F Connection handle  
HSUB DS F Subscription handle  
ACTION DS F Action requested by MQSUBRQ  
SUBRQOPTS CMQSROA , Options that control the action of MQSUBRQ  
COMPCODE DS F Completion code  
REASON DS F Reason code qualifying COMPCODE
```

## Attributs des objets

Cette collection de rubriques répertorie uniquement les objets IBM MQ qui peuvent faire l'objet d'un appel de fonction MQINQ et fournit des détails sur les attributs pouvant être recherchés et les sélecteurs à utiliser.

### Attributs du gestionnaire de files d'attente

Certains attributs de gestionnaire de files d'attente sont fixes pour des implémentations particulières ; d'autres peuvent être modifiés à l'aide de la commande MQSC ALTER QMGR.

Les attributs peuvent également être affichés à l'aide de la commande DISPLAY QMGR. La plupart des attributs de gestionnaire de files d'attente peuvent être renseignés en ouvrant un objet MQOT\_Q\_MGR spécial et en utilisant l'appel MQINQ avec le descripteur renvoyé.

Le tableau suivant récapitule les attributs spécifiques au gestionnaire de files d'attente. Les attributs sont décrits par ordre alphabétique.

**Remarque :** Les noms des attributs affichés dans cette section sont des noms descriptifs utilisés avec l'appel MQINQ ; les noms sont identiques à ceux des commandes PCF. Lorsque des commandes MQSC sont utilisées pour définir, modifier ou afficher des attributs, d'autres noms abrégés sont utilisés ; voir Commandes MQSC pour plus d'informations.

Tableau 557. Attributs du gestionnaire de files d'attente	
Attribut	Description
<a href="#">AccountingConnOverride</a>	Remplacer les paramètres de comptabilité.
<a href="#">AccountingInterval</a>	Fréquence d'écriture des enregistrements comptables intermédiaires.
<a href="#">ActivityConnOverride</a>	Remplacer les paramètres d'activité.
<a href="#">ActivityTrace</a>	Contrôle la collecte de la trace de l'activité de l'application IBM MQ MQI.
<a href="#">AdoptNewMCACheck</a>	Éléments vérifiés pour déterminer s'il convient d'adopter un nouvel agent MCA.
<a href="#">AdoptNewMCAType</a>	Indique s'il faut redémarrer automatiquement une instance orpheline d'un agent MCA d'un type de canal particulier.
<a href="#">AlterationDate</a>	Date de la dernière modification de la définition
<a href="#">AlterationTime</a>	Heure de la dernière modification de la définition
<a href="#">AuthorityEvent</a>	Contrôle si des événements d'autorisation (non autorisés) sont générés
<a href="#">BridgeEvent</a>	Attribut de contrôle pour les événements de pont.
<a href="#">ChannelAutoDef</a>	Contrôle si la définition de canal automatique est autorisée
<a href="#">ChannelAutoDefEvent</a>	Contrôle si des événements de définition automatique de canal sont générés
<a href="#">ChannelAutoDefExit</a>	Nom de l'exit utilisateur pour la définition de canal automatique
<a href="#">ChannelEvent</a>	Attribut de contrôle pour les événements de canal.
<a href="#">ChannelInitiatorControl</a>	Attribut de contrôle de l'initiateur de canal
<a href="#">ChannelMonitoring</a>	Données de surveillance en ligne pour les canaux
<a href="#">ChannelStatistics</a>	Contrôle la collecte des données statistiques pour les canaux.
<a href="#">ChinitAdapters</a>	Nombre de sous-tâches d'adaptateur pour le traitement des appels IBM MQ .
<a href="#">ChinitDispatchers</a>	Nombre de répartiteurs à utiliser pour l'initiateur de canal.
	Réservé à l'utilisation par IBM .
<a href="#">ChinitTraceAutoStart</a>	Indique si la trace de l'initiateur de canal doit démarrer automatiquement.
<a href="#">ChinitTraceTableSize</a>	Taille de l'espace de données de trace de l'initiateur de canal.
<a href="#">ClusterSenderMonitoringDefault</a>	Valeurs par défaut des données de surveillance en ligne pour les canaux émetteurs de cluster
<a href="#">StatistiquesClusterSender</a>	Contrôle la collecte des informations de surveillance des statistiques pour les canaux émetteurs de cluster.
<a href="#">ClusterWorkloadData</a>	Données utilisateur pour l'exit de charge de travail de cluster
<a href="#">ClusterWorkloadExit</a>	Nom de l'exit utilisateur pour la gestion de la charge de travail du cluster
<a href="#">ClusterWorkloadLength</a>	Longueur maximale des données de message transmises à l'exit de charge de travail du cluster
<a href="#">CLWLMRUChannels</a>	Nombre de canaux les plus récemment utilisés pour l'équilibrage de charge de cluster
<a href="#">CLWLUseQ</a>	La charge de travail du cluster utilise la file d'attente éloignée.
<a href="#">CodedCharSetId</a>	Identificateur de jeu de caractères codés
<a href="#">CommandEvent</a>	Attribut de contrôle pour les événements de commande.
<a href="#">AttributCommandInputQName</a>	Nom de la file d'attente d'entrée des commandes
<a href="#">CommandLevel</a>	Niveau de commande
<a href="#">AttributCommandServerControl</a>	Attribut de contrôle du serveur de commandes.
<a href="#">Attribut Événement de configuration</a>	Attribut de contrôle pour les événements de configuration.
<a href="#">DeadLetterQName</a>	Nom de la file d'attente de rebut
<a href="#">DefClusterXmitQueue</a>	Type de file d'attente de transmission de cluster par défaut

Tableau 557. Attributs du gestionnaire de files d'attente (suite)



Attribut	Description
<a href="#">DefXmitQName</a>	Nom de file d'attente de transmission par défaut
<a href="#">DistLists</a>	Prise en charge de la liste de diffusion
<a href="#">DNSGroup</a>	Nom du groupe pour le programme d'écoute TCP lors de l'utilisation de la prise en charge de Workload Manager Dynamic Domain Name Services.
<a href="#">DNSWLM</a>	Indique si le programme d'écoute TCP s'enregistre auprès de Workload Manager pour Dynamic Domain Name Services.
<a href="#">ExpiryInterval</a>	Intervalle entre les analyses des messages arrivés à expiration
<a href="#">IGQPutAuthority</a>	Droit d'insertion de la mise en file d'attente intragroupe
<a href="#">IGQUserId</a>	Identificateur utilisateur de la mise en file d'attente intra-groupe
<a href="#">InhibitEvent</a>	Contrôle si des événements d'inhibition (Inhibition Get et Inhibition Put) sont générés
 InitialKey 1	Clé initiale du système de protection par mot de passe.
<a href="#">IPAddressVersion</a>	Version de l'adresse Internet Protocol
<a href="#">IntraGroupqueuing</a>	Prise en charge de la mise en file d'attente intra-groupe
<a href="#">ListenerTimer</a>	Intervalle entre les tentatives de redémarrage du programme d'écoute après un échec APPC ou TCP/IP.
<a href="#">LocalEvent</a>	Contrôle si des événements d'erreur locaux sont générés
<a href="#">LoggerEvent</a>	Contrôle si les événements du consignateur sont générés
<a href="#">LUGroupName</a>	Nom de LU générique du programme d'écoute de LU 6.2 qui gère les transmissions entrantes pour le groupe de partage de files d'attente.
<a href="#">nom ul</a>	Nom de l'unité logique à utiliser pour les transmissions LU 6.2 sortantes.
<a href="#">LU62ARMSuffix</a>	Suffixe de SYS1.PARMLIB APPCPMxx, qui désigne LUADD pour cet initiateur de canal.
<a href="#">LU62Channels</a>	Nombre maximal de canaux en cours ou de clients connectés qui utilisent l'unité logique 6.2.
<a href="#">MaxActiveChannels</a>	Nombre maximal de canaux pouvant être actifs à tout moment.
<a href="#">MaxChannels</a>	Nombre maximal de canaux en cours.
<a href="#">MaxHandles</a>	Nombre maximal de descripteurs
<a href="#">Longueur maximale des messages</a>	Longueur maximale des messages (en octets)
<a href="#">AttributMaxPriority</a>	Priorité maximum
<a href="#">MaxPropertiesLength</a>	Longueur maximale des données de propriété en octets
<a href="#">MaxUncommittedMsgs</a>	Nombre maximal de messages non validés dans une unité de travail
<a href="#">MQIAccounting</a>	Contrôle la collecte des informations de comptabilité pour les données MQI.
<a href="#">MQIStatistics</a>	Contrôle la collecte des informations de surveillance des statistiques pour le gestionnaire de files d'attente.
<a href="#">MsgMarkBrowseInterval</a>	Intervalle après lequel le gestionnaire de files d'attente peut supprimer la marque des messages consultés.
<a href="#">OutboundPortMin</a>	Avec <i>OutboundPortMin</i> , définit une plage de numéros de port à utiliser lors de la liaison des canaux sortants.
<a href="#">OutboundPortMax</a>	Avec <i>OutboundPortMax</i> , définit une plage de numéros de port à utiliser lors de la liaison des canaux sortants.
<a href="#">PerformanceEvent</a>	Contrôle si les événements liés aux performances sont générés
<a href="#">Plateforme</a>	Plateforme sur laquelle le gestionnaire de files d'attente est en cours d'exécution
<a href="#">PubSubNPInputMsg</a>	Indique s'il faut supprimer (ou conserver) un message d'entrée non distribué
<a href="#">PubSubNPResponse</a>	Contrôle le comportement des commandes non livrées
<a href="#">PubSubMaxMsgRetryCount</a>	Nombre de nouvelles tentatives lors du traitement (sous le point de synchronisation) d'un message de commande ayant échoué
<a href="#">PubSubSyncPoint</a>	Indique si seuls les messages persistants (ou tous les messages) doivent être traités pour un point de synchronisation.
<a href="#">PubSubMode</a>	Indique si l'interface de publication / abonnement en file d'attente est en cours d'exécution
<a href="#">QMGrDesc</a>	Description du gestionnaire de files d'attente
<a href="#">QMGrIdentifier</a>	Identificateur unique généré en interne du gestionnaire de files d'attente

Tableau 557. Attributs du gestionnaire de files d'attente (suite)

Attribut	Description
<a href="#">QMgrName</a>	Nom gest. de files
<a href="#">QSGName</a>	Nom du groupe de partage de files d'attente
<a href="#">QueueAccounting</a>	Contrôle la collecte des informations comptables pour les files d'attente.
<a href="#">QueueMonitoring</a>	Données de surveillance en ligne pour les files d'attente
<a href="#">QueueStatistics</a>	Contrôle la collecte des données statistiques pour les files d'attente.
<a href="#">ReceiveTimeout</a>	Durée pendant laquelle le canal TCP/IP attend les données avant de revenir à l'état inactif.
<a href="#">ReceiveTimeoutMin</a>	Qualificateur pour <i>ReceiveTimeout</i> .
<a href="#">ReceiveTimeoutType</a>	Durée minimale pendant laquelle le canal TCP/IP attend les données avant de revenir à l'état inactif.
<a href="#">RemoteEvent</a>	Contrôle si des événements d'erreur distants sont générés
<a href="#">RepositoryName</a>	Nom du cluster pour lequel ce gestionnaire de files d'attente fournit des services de référentiel
<a href="#">RepositoryNamelist</a>	Nom de l'objet liste de noms contenant les noms des clusters pour lesquels ce gestionnaire de files d'attente fournit des services de référentiel
<a href="#">ScyCase</a>	Cas des profils de sécurité
<a href="#">SharedQMgr</a>	Nom du gestionnaire de files d'attente partagées
«SPLCAP», à la page 871	IBM MQ Protection avancée de la sécurité des messages pour un gestionnaire de files d'attente activée ou désactivée.
<a href="#">SSLCRLNamelist 1</a>	Nom de l'objet liste de noms contenant les noms des objets d'informations d'authentification.
<a href="#">SSLCryptoHardware 1</a>	Chaîne de configuration du matériel de cryptographie.
<a href="#">SSLEvent</a>	Attribut de contrôle pour les événements TLS.
<a href="#">SSLFIPSRequired</a>	Utilisez uniquement des algorithmes certifiés FIPS pour la cryptographie.
<a href="#">SSLKeyRepository 1</a>	Emplacement du référentiel de clés TLS.
 <a href="#">SSLKeyRepositoryMot de passe 1</a>	Mot de passe du référentiel de clés TLS.
<a href="#">SSLKeyResetNombre</a>	Nombre de réinitialisations de clé TLS.
<a href="#">SSLTpose 1</a>	Nombre de sous-tâches de serveur pour le traitement des appels TLS.
<a href="#">StatisticsInterval</a>	Fréquence d'écriture des données de surveillance des statistiques.
<a href="#">StartStopEvent</a>	Contrôle si des événements de démarrage et d'arrêt sont générés
<a href="#">SyncPoint</a>	Prise en charge points sync.
<a href="#">TCPChannels</a>	Nombre maximal de canaux en cours ou de clients connectés qui utilisent TCP/IP.
<a href="#">TCPKeepAlive</a>	Indique si TCP KEEPALIVE doit être utilisé pour vérifier les autres fins de connexion.
<a href="#">TCPName</a>	Nom du système TCP/IP que vous utilisez.
<a href="#">TCPStackType</a>	Comment l'initiateur de canal peut utiliser les adresses TCP/IP.
<a href="#">AttributTraceRouteEnregistrement</a>	Contrôle l'enregistrement des informations de routage de trace.
<a href="#">TriggerInterval</a>	Intervalle entre les messages de déclenchement
<a href="#">Version</a>	Version
<a href="#">XrCapability</a>	Indique si les commandes de télémétrie sont prises en charge.
<b>Remarques :</b>	
1. Cet attribut ne peut pas être interrogé à l'aide de l'appel MQINQ et n'est pas décrit dans cette section. Pour plus d'informations sur cet attribut, voir <a href="#">Modification du gestionnaire de files d'attente</a> .	

**Tâches associées**

Comment indiquer que seuls les CipherSpecs certifiés FIPS sont utilisés lors de l'exécution sur MQI Client

**Référence associée**

FIPS (Federal Information Processing Standards) pour AIX, Linux, and Windows

## **Remplacement de AccountingConn(MQLONG)**

Cela permet aux applications de remplacer le paramètre des valeurs ACCTMQI et ACCTQDATA dans l'attribut Qmgr.

La valeur est l'une des suivantes :

### **MQMON\_DISABLED**


Les applications ne peuvent pas remplacer les paramètres des attributs ACCTMQI et ACCTQ Qmgr à l'aide de la zone Options de la structure MQCNO sur l'appel MQCONN. Il s'agit de la valeur par défaut.

### **MQMON\_ENABLED**

Les applications peuvent remplacer les attributs Qmgr ACCTQ et ACCTMQI à l'aide de la zone Options de la structure MQCNO.

Les modifications apportées à cette valeur ne sont effectives que pour les connexions au gestionnaire de files d'attente après la modification de l'attribut.

Cet attribut est pris en charge uniquement sur les plateformes suivantes:

-  IBM i
-   AIX and Linux
-  Windows

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_ACCOUNTING\_CONN\_OVERRIDE avec l'appel MQINQ.

## **AccountingInterval (MQLONG)**

Indique la durée d'écriture des enregistrements comptables intermédiaires (en secondes).

La valeur est un entier compris entre 0 et 604800, avec une valeur par défaut de 1800 (30 minutes). Indiquez 0 pour désactiver les enregistrements intermédiaires.

Cet attribut est pris en charge uniquement sur les plateformes suivantes:

-  IBM i
-   AIX and Linux
-  Linux
-  Windows

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_ACCOUNTING\_INTERVAL avec l'appel MQINQ.

## **Substitution ActivityConn(MQLONG)**

Cela permet aux applications de remplacer la valeur ACTVTRC dans l'attribut de gestionnaire de files d'attente.

La valeur est l'une des suivantes :

### **MQMON\_DISABLED**

Les applications ne peuvent pas remplacer le paramètre de l'attribut de gestionnaire de files d'attente ACTVTRC à l'aide de la zone Options de la structure MQCNO sur l'appel MQCONN. Il s'agit de la valeur par défaut.

### **MQMON\_ENABLED**

Les applications peuvent remplacer l'attribut de gestionnaire de files d'attente ACTVTRC à l'aide de la zone Options de la structure MQCNO.

Les modifications apportées à cette valeur ne sont effectives que pour les connexions au gestionnaire de files d'attente après la modification de l'attribut.

Cet attribut est pris en charge uniquement sous [Multiplateformes](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_ACTIVITY\_CONN\_OVERRIDE avec l'appel MQINQ .

### ***ActivityTrace (MQLONG)***

Cette option contrôle la collecte de la trace de l'activité de l'application IBM MQ MQI.

La valeur est l'une des suivantes :

#### **MQMON\_ON**

Collectez la trace de l'activité de l'application IBM MQ MQI.

#### **MQMON\_OFF**

Ne collectez pas de trace d'activité d'application IBM MQ MQI. Il s'agit de la valeur par défaut.

Si vous définissez l'attribut de gestionnaire de files d'attente ACTVCON0 sur ENABLED, cette valeur peut être remplacée pour des connexions individuelles à l'aide de la zone Options de la structure MQCNO.

Les modifications apportées à cette valeur ne sont effectives que pour les connexions au gestionnaire de files d'attente après la modification de l'attribut.

Cet attribut est pris en charge uniquement sous [Multiplateformes](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_ACTIVITY\_TRACE avec l'appel MQINQ .

### ***AdoptNewMCACheck (MQLONG)***

Définit les éléments à vérifier pour déterminer s'il convient d'adopter un agent MCA lorsqu'un nouveau canal entrant portant le même nom qu'un agent MCA déjà actif est détecté.

La valeur est l'une des suivantes :

#### **MQADOPT\_CHECK\_Q\_MGR\_NAME**

Vérifiez le nom du gestionnaire de files d'attente.

#### **MQADOPT\_CHECK\_NET\_ADDR**

Vérifiez l'adresse réseau.


#### **MQADOPT\_CHECK\_ALL**

Vérifiez le nom du gestionnaire de files d'attente et l'adresse réseau. Si possible, effectuez cette vérification pour empêcher l'arrêt de vos canaux, par inadvertance ou de manière malveillante. Il s'agit de la valeur par défaut.

#### **MQADOPT\_CHECK\_NONE**

Ne cochez aucun élément.

Les modifications apportées à cet attribut prennent effet la prochaine fois qu'un canal tente d'adopter un canal.

 Cet attribut est pris en charge uniquement sous z/OS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_ADOPTNEWMCA\_CHECK avec l'appel MQINQ.

### ***AdoptNewMCAType (MQLONG)***

Indique s'il faut redémarrer automatiquement une instance orpheline d'un agent MCA d'un type de canal particulier lorsqu'une nouvelle demande de canal entrant correspondant à l'attribut MCACheck AdoptNewest détectée

Ses valeurs sont l'une des suivantes :



**MQADOPT\_TYPE\_NO**

L'adoption d'instances de canal orphelines n'est pas requise. Il s'agit de la valeur par défaut.

**MQADOPT\_TYPE\_ALL**

Adoptez tous les types de canal.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_ADOPTNEWMCA\_TYPE avec l'appel MQINQ.

**AlterationDate (MQCHAR12)**

Il s'agit de la date de la dernière modification de la définition. Le format de la date est YYYY-MM-DD, complété par deux blancs de fin pour que la longueur soit de 12 octets.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_ALTERATION\_DATE avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_DATE\_LENGTH.

**AlterationTime (MQCHAR8)**

Il s'agit de l'heure à laquelle la définition a été modifiée pour la dernière fois. Le format de l'heure est HH.MM.SS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_ALTERATION\_TIME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_TIME\_LENGTH.

**AuthorityEvent (MQLONG)**

Indique si les événements d'autorisation (Non autorisé) sont générés. Ses valeurs sont l'une des suivantes :

**MQEVR\_DISABLED**

Génération de rapports d'événements désactivée.

**MQEVR\_ENABLED**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_AUTHORITY\_EVENT avec l'appel MQINQ.

**BridgeEvent (MQLONG)**

Indique si des événements de pont IMS sont générés.

La valeur est l'une des suivantes :

**MQEVR\_ENABLED**

Générez des événements de pont IMS , comme suit:

MQRC\_BRIDGE\_STARTED

MQRC\_BRIDGE\_STOPPED

**MQEVR\_DISABLED**

Ne générez pas d'événements de pont IMS ; il s'agit de la valeur par défaut.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_BRIDGE\_EVENT avec l'appel MQINQ.

**Def ChannelAuto(MQLONG)**


Cet attribut contrôle la définition automatique des canaux de type MQCHT\_RECEIVER et MQCHT\_SVRCONN. La définition automatique des canaux MQCHT\_CLUSSDR est toujours activée. La valeur est l'une des suivantes :

## **MQCHAD\_DISABLED**

Définition automatique de canal désactivée.

## **MQCHAD\_ENABLED**

Définition automatique de canal activée.

 Cet attribut est pris en charge uniquement sous [Multiplateformes](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_CHANNEL\_AUTO\_DEF avec l'appel MQINQ.

## ***ChannelAutoDefEvent (MQLONG)***

Indique si les événements de définition automatique de canal sont générés. Elle s'applique aux canaux de type MQCHT\_RECEIVER, MQCHT\_SVRCONN et MQCHT\_CLUSSDR. La valeur est l'une des suivantes :

### **MQEVR\_DISABLED**

Génération de rapports d'événements désactivée.

### **MQEVR\_ENABLED**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

 Cet attribut est pris en charge uniquement sous [Multiplateformes](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_CHANNEL\_AUTO\_DEF\_EVENT avec l'appel MQINQ.

## ***ChannelAutoDefExit (MQCHARn)***

Il s'agit du nom de l'exit utilisateur pour la définition de canal automatique. Si ce nom n'est pas vide et que *ChannelAutoDef* a la valeur MQCHAD\_ENABLED, l'exit est appelé chaque fois que le gestionnaire de files d'attente est sur le point de créer une définition de canal. S'applique aux canaux de type MQCHT\_RECEIVER, MQCHT\_SVRCONN et MQCHT\_CLUSSDR. L'exit peut alors effectuer l'une des opérations suivantes:

- Créez la définition de canal sans modification.
- Modifiez les attributs de la définition de canal créée.
- Supprimer complètement la création du canal.

**Remarque :** La longueur et la valeur de cet attribut sont spécifiques à l'environnement. Pour plus de détails sur la valeur de cet attribut dans divers environnements, voir l'introduction de la structure MQCD dans «MQCD-Définition de canal», à la page 1553 .

 Sous z/OS, cet attribut s'applique uniquement aux canaux émetteur et récepteur de cluster.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_CHANNEL\_AUTO\_DEF\_EXIT avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_EXIT\_NAME\_LENGTH.

## ***ChannelEvent (MQLONG)***

Indique si des événements de canal sont générés.

Ses valeurs sont l'une des suivantes :

### **MQEVR\_EXCEPTION**

Générez uniquement les événements de canal suivants:

- MQRC\_CHANNEL\_ACTIVATED
- MQRC\_CHANNEL\_CONV\_ERROR
- MQRC\_CHANNEL\_NOT\_ACTIVATED
- MQRC\_CHANNEL\_STOPPED avec les ReasonQualifiers suivants:

MQRQ\_CHANNEL\_STOPPED\_ERROR  
MQRQ\_CHANNEL\_STOPPED\_RETRY  
MQRQ\_CHANNEL\_STOPPED\_DISABLED

MQRC\_CHANNEL\_STOPPED\_BY\_USER

#### **MQEVN\_ENABLED**

Générez tous les événements de canal. En d'autres termes, en plus de celles générées par EXCEPTION, générez les événements de canal suivants:

- MQRC\_CHANNEL\_STARTED
- MQRC\_CHANNEL\_STOPPED avec l'élément ReasonQualifiers suivant:

MQRQ\_CHANNEL\_STOPPED\_OK

#### **MQEVN\_DISABLED**

Ne générez pas d'événements de canal ; il s'agit de la valeur par défaut.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_CHANNEL\_EVENT avec l'appel MQINQ.

#### **Contrôle Channel Initiator (MQLONG)**

Indique si l'initiateur de canal doit être démarré lors du démarrage du gestionnaire de files d'attente.

Ses valeurs sont l'une des suivantes :

#### **MQSVC\_CONTROL\_MANUAL**

L'initiateur de canal ne doit pas être démarré automatiquement.

#### **MQSVC\_CONTROL\_Q\_MGR**

L'initiateur de canal doit être démarré automatiquement au démarrage du gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_CHINIT\_CONTROL avec l'appel MQINQ.

#### **Channel Monitoring (MQLONG)**

Cet attribut spécifie les données de surveillance en ligne pour les canaux.

La valeur est l'une des suivantes :

#### **MQMON\_AUCUN**

Désactivez la collecte de données pour la surveillance des canaux pour tous les canaux, quelle que soit la valeur de l'attribut de canal MONCHL. Il s'agit de la valeur par défaut.

#### **MQMON\_OFF**

Désactivez la collecte des données de surveillance pour les canaux qui spécifient QMGR dans l'attribut de canal MONCHL.

#### **MQMON\_FAIBLE**


Activez la collecte de données de surveillance avec un faible taux de collecte de données pour les canaux spécifiant QMGR dans l'attribut de canal MONCHL.

#### **MQMON\_MEDIUM**

Activez la collecte de données de surveillance avec un rapport modéré de collecte de données pour les canaux spécifiant QMGR dans l'attribut de canal MONCHL.

#### **MQMON\_ELEVE**

Activez la collecte de données de surveillance avec un taux élevé de collecte de données pour les canaux spécifiant QMGR dans l'attribut de canal MONCHL.

 Sur les systèmes z/OS, l'activation de ce paramètre active simplement la collecte de données statistiques, quelle que soit la valeur que vous sélectionnez. La spécification de LOW, MEDIUM ou HIGH n'a pas d'impact sur vos résultats.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_MONITORING\_CHANNEL avec l'appel MQINQ.

### ***ChannelStatistics (MQLONG)***

Cette option contrôle la collecte des données statistiques pour les canaux.

La valeur est l'une des suivantes :

#### **MQMON\_AUCUN**

Désactivez la collecte de données pour les statistiques de canal pour tous les canaux, quelle que soit la valeur de l'attribut de canal STATCHL. Il s'agit de la valeur par défaut.

#### **MQMON\_OFF**

Désactivez la collecte des données statistiques pour les canaux qui spécifient QMGR dans l'attribut de canal STATCHL.

#### **MQMON\_FAIBLE**

Activez la collecte de données statistiques avec un faible taux de collecte de données pour les canaux spécifiant QMGR dans l'attribut de canal STATCHL.


#### **MQMON\_MEDIUM**

Activez la collecte de données statistiques avec un taux modéré de collecte de données pour les canaux spécifiant QMGR dans l'attribut de canal STATCHL.

#### **MQMON\_ELEVE**

Activez la collecte de données statistiques avec un taux élevé de collecte de données pour les canaux spécifiant QMGR dans l'attribut de canal STATCHL.

Pour la plupart des systèmes, il est recommandé d'utiliser MEDIUM. Toutefois, pour un canal qui traite un volume élevé de messages par seconde, vous pouvez réduire le niveau d'échantillonnage en sélectionnant LOW. En outre, pour un canal qui ne traite que quelques messages et pour lequel les informations les plus récentes sont importantes, vous pouvez sélectionner HIGH.

 Sur les systèmes z/OS, l'activation de ce paramètre active simplement la collecte de données statistiques, quelle que soit la valeur que vous sélectionnez. La spécification de LOW, MEDIUM ou HIGH n'a pas d'impact sur vos résultats. Vous devez activer ce paramètre pour pouvoir collecter des enregistrements de comptabilité de canal.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_STATISTICS\_CHANNEL avec l'appel MQINQ.

### ***ChinitAdapters (MQLONG)***

Il s'agit du nombre de sous-tâches d'adaptateur à utiliser pour traiter les appels IBM MQ. La valeur doit être comprise entre 0 et 9999, avec une valeur par défaut de 8.

Le rapport entre les adaptateurs et les répartiteurs (attribut ChinitDispatchers) doit être d'environ 8 à 5. Toutefois, si vous n'avez que quelques canaux, vous n'avez pas besoin de diminuer la valeur de ce paramètre par rapport à la valeur par défaut. Vous pouvez utiliser les valeurs suivantes: pour un système de test, 8 (valeur par défaut); pour un système de production, 20. Dans l'idéal, vous devez disposer de 20 adaptateurs, ce qui permet un plus grand parallélisme des appels IBM MQ. Cette caractéristique est importante pour les messages persistants. Moins d'adaptateurs peuvent être utilisés pour les messages non persistants.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_CHINIT\_ADAPTERS avec l'appel MQINQ.

### ***ChinitDispatchers (MQLONG)***

Nombre de répartiteurs à utiliser pour l'initiateur de canal. La valeur doit être comprise entre 0 et 9999, avec une valeur par défaut de 5.

Comme instruction, autorisez un répartiteur pour 50 canaux en cours. Toutefois, si vous n'avez que quelques canaux, vous n'avez pas besoin de diminuer la valeur de cet attribut par rapport à la valeur par défaut. Si vous utilisez TCP/IP, le plus grand nombre de répartiteurs utilisés pour les canaux TCP/IP est 100, même si vous indiquez une valeur plus élevée ici. Vous pouvez utiliser les paramètres suivants: systèmes de test, 5 (par défaut) ; systèmes de production, 20 (vous avez besoin de 20 répartiteurs pour gérer jusqu'à 1000 canaux actifs).

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_CHINIT\_DISPATCHERS avec l'appel MQINQ.

### ***ChinitTraceAutoStart (MQLONG)***

Indique si la trace de l'initiateur de canal doit être démarrée automatiquement.

La valeur est l'une des suivantes :

#### **MQTRAXSTR\_YES**

Démarrer automatiquement la trace de l'initialisateur de canal. Il s'agit de la valeur par défaut.

#### **MQTRAXSTR\_NO**

Ne pas démarrer automatiquement la trace de l'initialisateur de canal.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_CHINIT\_TRACE\_AUTO\_START avec l'appel MQINQ.

### ***ChinitTraceTableSize (MQLONG)***

Il s'agit de la taille de l'espace de données de trace de l'initiateur de canal (en Mo).

La valeur doit être comprise entre 0 et 2048, avec une valeur par défaut de 2.

**Remarque :** Chaque fois que vous utilisez des espaces de données z/OS volumineux, vérifiez que vous disposez de suffisamment de mémoire secondaire sur votre système pour prendre en charge toute activité de pagination z/OS associée. Vous devrez peut-être également augmenter la taille de vos fichiers SYS1.DUMP.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_CHINIT\_TRACE\_TABLE\_SIZE avec l'appel MQINQ.

### ***ClusterSenderMonitoringDefault (MQLONG)***

Indique la valeur à remplacer par l'attribut ChannelMonitoring des canaux émetteurs de cluster définis automatiquement.

La valeur est l'une des suivantes :

#### **MQMON\_Q\_DIR**

La collecte des données de surveillance en ligne est héritée de la valeur de l'attribut **ChannelMonitoring** du gestionnaire de files d'attente. Il s'agit de la valeur par défaut.

#### **MQMON\_OFF**

La surveillance du canal est désactivée

#### **MQMON\_FAIBLE**

A moins que *ChannelMonitoring* ne soit MQMON\_NONE, la surveillance est activée avec un faible taux de collecte de données avec un impact minimal sur les performances du système. Il est peu probable que les données collectées soient les plus récentes.

#### **MQMON\_MEDIUM**

A moins que *ChannelMonitoring* ne soit MQMON\_NONE, la surveillance est activée avec un taux modéré de collecte de données avec un effet limité sur les performances du système.

## **MQMON\_ELEVE**

A moins que *ChannelMonitoring* ne soit défini sur MQMON\_NONE, la surveillance est activée avec un taux élevé de collecte de données, ce qui a un impact probable sur les performances du système. Les données collectées sont les plus récentes disponibles.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_MONITORING\_AUTO\_CLUSSDR avec l'appel MQINQ.

## **ClusterSender-Statistiques (MQLONG)**

Etant donné que les canaux émetteurs de cluster peuvent être définis automatiquement à partir de la définition de CLUSRCVR dans le référentiel, vous ne pouvez pas modifier le paramètre de l'attribut STATCHL pour ces canaux émetteurs de cluster définis automatiquement à l'aide du canal ALTER. Pour ces canaux, la décision de collecter ou non des données de surveillance en ligne est basée sur la définition de cet attribut de gestionnaire de files d'attente.

La valeur est l'une des suivantes :

### **MQMON\_Q\_DIR**

La collecte de données statistiques pour les canaux émetteurs de cluster définis automatiquement est basée sur la valeur de l'attribut de gestionnaire de files d'attente STATCHL. Il s'agit de la valeur par défaut.

### **MQMON\_OFF**

Désactivez la collecte des données statistiques pour les canaux émetteurs de cluster définis automatiquement.

### **MQMON\_FAIBLE**

Activez la collecte de données statistiques pour les canaux émetteurs de cluster définis automatiquement avec un faible taux de collecte de données.


### **MQMON\_MEDIUM**

Activez la collecte de données statistiques pour les canaux émetteurs de cluster définis automatiquement avec un taux modéré de collecte de données.

### **MQMON\_ELEVE**

Activez la collecte de données statistiques pour les canaux émetteurs de cluster définis automatiquement avec un taux élevé de collecte de données.

Pour la plupart des systèmes, nous recommandons MEDIUM. Toutefois, pour un canal émetteur de cluster défini automatiquement qui traite un volume élevé de messages par seconde, vous pouvez réduire le niveau d'échantillonnage en sélectionnant LOW. En outre, pour un canal qui ne traite que quelques messages et pour lequel les informations les plus récentes sont importantes, vous pouvez sélectionner HIGH.

 Sur les systèmes z/OS, l'activation de ce paramètre active simplement la collecte de données statistiques, quelle que soit la valeur que vous sélectionnez. La spécification de LOW, MEDIUM ou HIGH n'a pas d'impact sur vos résultats. Vous devez activer ce paramètre pour pouvoir collecter des enregistrements de comptabilité de canal.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_STATISTICS\_AUTO\_CLUSSDR avec l'appel MQINQ.

## **ClusterWorkload-Données (MQCHAR32)**

Il s'agit d'une chaîne de caractères de 32 octets définie par l'utilisateur qui est transmise à l'exit de charge de travail du cluster lorsqu'il est appelé. S'il n'y a pas de données à transmettre à l'exit, la chaîne est vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_CLUSTER\_WORKLOAD\_DATA avec l'appel MQINQ.

## **Exit ClusterWorkload(MQCHARn)**

Il s'agit du nom de l'exit utilisateur pour la gestion de la charge de travail du cluster. Si ce nom n'est pas vide, l'exit est appelé chaque fois qu'un message est inséré dans une file d'attente de cluster ou déplacé d'une file d'attente d'émetteur de cluster vers une autre. L'exit peut alors soit accepter l'instance de file d'attente sélectionnée par le gestionnaire de files d'attente comme destination du message, soit sélectionner une autre instance de file d'attente.

**Remarque :** La longueur et la valeur de cet attribut sont spécifiques à l'environnement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_CLUSTER\_WORKLOAD\_EXIT avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_EXIT\_NAME\_LENGTH.

### ***ClusterWorkloadLongueur (MQLONG)***

Longueur maximale des données de message transmises à l'exit de charge de travail du cluster. La longueur réelle des données transmises à l'exit est la valeur minimale suivante:

- Longueur du message.
- Attribut **MaxMsgLength** du gestionnaire de files d'attente.
- Attribut **ClusterWorkloadLength**.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_CLUSTER\_WORKLOAD\_LENGTH avec l'appel MQINQ.

### ***CLWLMRUChannels (MQLONG)***

Indique le nombre maximal de canaux de cluster les plus récemment utilisés, à utiliser par l'algorithme de choix de charge de travail de cluster.

Il s'agit d'une valeur comprise entre 1 et 999999999.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_CLWL\_MRU\_CHANNELS avec l'appel MQINQ.

### ***CLWLUseQ (MQLONG)***

Indique si les files d'attente éloignées doivent être utilisées pour la charge de travail du cluster.

La valeur est l'une des suivantes :

#### **MQCLWL\_USEQ\_ANY**

Utilisez les files d'attente locales et distantes.

#### **MQCLWL\_USEQ\_LOCAL**

N'utilisez pas de files d'attente éloignées. Il s'agit de la valeur par défaut.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_CLWL\_USEQ avec l'appel MQINQ.

### ***CodedCharSetId (MQLONG)***

Définit le jeu de caractères utilisé par le gestionnaire de files d'attente pour toutes les zones de chaîne de caractères définies dans l'interface MQI, telles que les noms des objets, ainsi que la date et l'heure de création de la file d'attente. Le jeu de caractères doit comporter des caractères mono-octet pour les caractères admis dans les noms d'objet. Elle ne s'applique pas aux données d'application contenues dans le message. La valeur dépend de l'environnement:

- Sous z/OS, la valeur est définie à partir des paramètres système lorsque le gestionnaire de files d'attente est démarré ; la valeur par défaut est 500.
- Sous Windows, la valeur est le CODEPAGE principal de l'utilisateur qui crée le gestionnaire de files d'attente.
- Sous IBM i, la valeur est celle qui est définie dans l'environnement lorsque le gestionnaire de files d'attente est créé pour la première fois.
- Sous AIX and Linux, la valeur est la valeur par défaut CODESET pour l'environnement local de l'utilisateur qui crée le gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_CODED\_CHAR\_SET\_ID avec l'appel MQINQ.

### **CommandEvent (MQLONG)**

Indique si des événements de commande sont générés, comme suit:

#### **MQEVR\_DISABLED**

Ne générez pas d'événements de commande. Il s'agit de l'option par défaut.

#### **MQEVR\_ENABLED**

Générez des événements de commande.

#### **MQEVR\_NO\_DISPLAY**

Des événements de commande sont générés pour toutes les commandes ayant abouti autres que MQINQ.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_COMMAND\_EVENT avec l'appel MQINQ.

### **CommandInputQName (MQCHAR48)**

Il s'agit du nom de la file d'attente d'entrée de commandes définie sur le gestionnaire de files d'attente local. Il s'agit d'une file d'attente à laquelle les utilisateurs peuvent envoyer des commandes, s'ils y sont autorisés. Le nom de la file d'attente dépend de l'environnement:

- Sous z/OS, le nom de la file d'attente est SYSTEM.COMMAND.INPUT; les commandes MQSC et PCF peuvent lui être envoyées. Voir [Les commandes MQSC](#) pour plus de détails sur les commandes MQSC et [Définitions des formats de commande programmables](#) pour plus de détails sur les commandes PCF.
- Dans tous les autres environnements, le nom de la file d'attente est SYSTEM.ADMIN.COMMAND.QUEUE, et seules les commandes PCF peuvent lui être envoyées. Toutefois, une commande MQSC peut être envoyée à cette file d'attente si la commande MQSC est incluse dans une commande PCF de type MQCMD\_ESCAPE. Pour plus d'informations sur la commande Escape, voir [Echap](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_COMMAND\_INPUT\_Q\_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_Q\_NAME\_LENGTH.

### **CommandLevel (MQLONG)**

**Remarque :** La prise en charge du système d'exploitation HP-UX pour tous les composants IBM MQ, y compris le serveur et les clients, a été supprimée dans IBM MQ 9.1.

Indique le niveau des commandes de contrôle du système prises en charge par le gestionnaire de files d'attente. Il peut s'agir de l'une des valeurs suivantes:

#### **MQCMDL\_LEVEL\_800**

Niveau 800 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 8.0
- IBM MQ for IBM i 8.0
- IBM MQ for Linux 8.0
- IBM MQ for Windows 8.0
- IBM MQ for z/OS 8.0

#### **MQCMDL\_LEVEL\_801**

Niveau 801 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 8.0.0 Fix Pack 2



- IBM MQ for HP-UX 8.0.0 Fix Pack 2
- IBM MQ for IBM i 8.0.0 Fix Pack 2
- IBM MQ for Linux 8.0.0 Fix Pack 2

#### **MQCMDL\_LEVEL\_802**

Niveau 802 des commandes de contrôle système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 8.0.0 Fix Pack 3
- IBM MQ for IBM i 8.0.0 Fix Pack 3
- IBM MQ for Linux 8.0.0 Fix Pack 3
- IBM MQ for Windows 8.0.0 Fix Pack 3

#### **MQCMDL\_LEVEL\_900**

Niveau 900 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 9.0
- IBM MQ for IBM i 9.0
- IBM MQ for Linux 9.0
- IBM MQ for Windows 9.0
- IBM MQ for z/OS 9.0

#### **MQCMDL\_LEVEL\_901**

Niveau 901 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for Linux 9.0.1
- IBM MQ for Windows 9.0.1
- IBM MQ for z/OS 9.0.1

#### **MQCMDL\_LEVEL\_902**

Niveau 902 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for Linux 9.0.2
- IBM MQ for Windows 9.0.2
- IBM MQ for z/OS 9.0.2

#### **MQCMDL\_LEVEL\_903**

Niveau 903 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for Linux 9.0.3
- IBM MQ for Windows 9.0.3
- IBM MQ for z/OS 9.0.3

#### **MQCMDL\_LEVEL\_904**

Niveau 904 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 9.0.4
- IBM MQ for Linux 9.0.4
- IBM MQ for Windows 9.0.4
- IBM MQ for z/OS 9.0.4

**MQCMDL\_LEVEL\_905**

Niveau 905 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 9.0.5
- IBM MQ for Linux 9.0.5
- IBM MQ for Windows 9.0.5
- IBM MQ for z/OS 9.0.5

**MQCMDL\_LEVEL\_910**

Niveau 910 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 9.1
- IBM MQ for IBM i 9.1
- IBM MQ for Linux 9.1
- IBM MQ for Windows 9.1
- IBM MQ for z/OS 9.1

**MQCMDL\_LEVEL\_911**

Niveau 911 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 9.1.1
- IBM MQ for Linux 9.1.1
- IBM MQ for Windows 9.1.1
- IBM MQ for z/OS 9.1.1

**MQCMDL\_LEVEL\_912**

Niveau 912 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 9.1.2
- IBM MQ for Linux 9.1.2
- IBM MQ for Windows 9.1.2
- IBM MQ for z/OS 9.1.2

**MQCMDL\_LEVEL\_913**

Niveau 913 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 9.1.3
- IBM MQ for Linux 9.1.3
- IBM MQ for Windows 9.1.3
- IBM MQ for z/OS 9.1.3

**MQCMDL\_LEVEL\_914**

Niveau 914 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 9.1.4
- IBM MQ for Linux 9.1.4
- IBM MQ for Windows 9.1.4
- IBM MQ for z/OS 9.1.4

**MQCMDL\_LEVEL\_915**

Niveau 915 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 9.1.5
- IBM MQ for Linux 9.1.5
- IBM MQ for Windows 9.1.5
- IBM MQ for z/OS 9.1.5

**MQCMDL\_LEVEL\_910**

Niveau 910 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 9.1
- IBM MQ for IBM i 9.1
- IBM MQ for Linux 9.1
- IBM MQ for Windows 9.1
- IBM MQ for z/OS 9.1

**MQCMDL\_LEVEL\_920**

Niveau 920 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 9.2
- IBM MQ for IBM i 9.2
- IBM MQ for Linux 9.2
- IBM MQ for Windows 9.2
- IBM MQ for z/OS 9.2

**MQCMDL\_LEVEL\_921**

Niveau 921 des commandes de contrôle système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 9.2.1
- IBM MQ for Linux 9.2.1
- IBM MQ for Windows 9.2.1
- IBM MQ for z/OS 9.2.1

**MQCMDL\_LEVEL\_922**

Niveau 922 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 9.2.2
- IBM MQ for Linux 9.2.2
- IBM MQ for Windows 9.2.2
- IBM MQ for z/OS 9.2.2

**MQCMDL\_LEVEL\_923**

Niveau 923 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 9.2.3
- IBM MQ for Linux 9.2.3
- IBM MQ for Windows 9.2.3

- IBM MQ for z/OS 9.2.3

#### **MQCMDL\_LEVEL\_924**

Niveau 924 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 9.2.4
- IBM MQ for Linux 9.2.4
- IBM MQ for Windows 9.2.4
- IBM MQ for z/OS 9.2.4

#### **MQCMDL\_LEVEL\_925**

Niveau 925 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 9.2.5
- IBM MQ for Linux 9.2.5
- IBM MQ for Windows 9.2.5
- IBM MQ for z/OS 9.2.5

#### **MQCMDL\_LEVEL\_930**

Niveau 930 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 9.3
- IBM MQ for IBM i 9.3
- IBM MQ for Linux 9.3
- IBM MQ for Windows 9.3
- IBM MQ for z/OS 9.3

#### **MQCMDL\_LEVEL\_931**

Niveau 931 des commandes de contrôle système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 9.3.1
- IBM MQ for Linux 9.3.1
- IBM MQ for Windows 9.3.1
- IBM MQ for z/OS 9.3.1

#### **MQCMDL\_LEVEL\_932**

Niveau 932 des commandes de contrôle système.

Cette valeur est renvoyée par les versions suivantes:

- IBM MQ for AIX 9.3.2
- IBM MQ for Linux 9.3.2
- IBM MQ for Windows 9.3.2
- IBM MQ for z/OS 9.3.2

L'ensemble de commandes de contrôle du système qui correspond à une valeur particulière de l'attribut **CommandLevel** varie en fonction de la valeur de l'attribut **Platform**. Ces deux commandes doivent être utilisées pour déterminer les commandes de contrôle du système qui sont prises en charge.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_COMMAND\_LEVEL avec l'appel MQINQ.

### **Contrôle CommandServer(MQLONG)**

Indique si le serveur de commandes doit être démarré lors du démarrage du gestionnaire de files d'attente.

La valeur peut être l'une des valeurs suivantes :

#### **MQSVC\_CONTROL\_MANUAL**

Le serveur de commandes ne doit pas être démarré automatiquement.

#### **MQSVC\_CONTROL\_Q\_MGR**

Le serveur de commandes doit être démarré automatiquement lors du démarrage du gestionnaire de files d'attente.

Cet attribut n'est pas pris en charge sous z/OS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_CMD\_SERVER\_CONTROL avec l'appel MQINQ.

### **ConfigurationEvent (MQLONG)**

Contrôle si les événements de configuration sont générés.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_CONFIGURATION\_EVENT avec l'appel MQINQ.

La valeur peut être l'une des valeurs suivantes :

#### **MQEVN\_DISABLED**

Génération de rapports d'événements désactivée.

#### **MQEVN\_ENABLED**

Génération de rapports d'événements activée.

### **Multi Taille CurrentQFile(MQLONG)**

Taille actuelle du fichier de file d'attente en mégaoctets, arrondie au mégaoctet le plus proche.

Tableau 558. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

La valeur de cet attribut de statut de file d'attente correspond à la taille actuelle de la file d'attente, arrondie au mégaoctet le plus proche. Pour une nouvelle file d'attente avec des attributs par défaut, la valeur de **CurrentQFileSize** est 1.

La valeur maximale de cet attribut est 99,999,9999 Mo et il n'existe pas de valeur par défaut pour cet attribut.

### **Multi CurrentMaxQFileSize (MQLONG)**

Taille maximale actuelle que le fichier de file d'attente peut atteindre, arrondie au mégaoctet le plus proche, compte tenu de la taille de bloc en cours d'utilisation dans une file d'attente.

Tableau 559. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

L'utilisation de ce champ est double:

- Si vous affectez à **MaxQFileSize** la valeur par défaut de la taille de bloc en cours, **CurrentMaxQFileSize** affiche la valeur réelle à laquelle la valeur par défaut correspond.
- Si **CurrentMaxQFileSize** ne correspond pas à **MaxQFileSize**, vous savez que la file d'attente doit être vidée afin d'adopter une granularité plus importante.

**Remarque :** Voir [Modification des fichiers de file d'attente IBM MQ](#) pour plus d'informations sur la modification de la taille des fichiers de file d'attente et de la taille et de la granularité des blocs.

La valeur maximale de cet attribut est 99,999,9999 Mo et il n'y a pas de valeur par défaut. La valeur correspond à la valeur maximale actuellement définie ; pour une nouvelle file d'attente avec les attributs par défaut, la valeur de **CurrentMaxQFileSize** est 2 088 960 Mo.

### **DeadLetterQName (MQCHAR48)**

Il s'agit du nom d'une file d'attente définie sur le gestionnaire de files d'attente local en tant que file d'attente de rebut (messages non livrés). Les messages sont envoyés à cette file d'attente s'ils ne peuvent pas être acheminés vers leur destination correcte.

Par exemple, les messages sont placés dans cette file d'attente lorsque:

- Un message arrive sur un gestionnaire de files d'attente, destiné à une file d'attente qui n'est pas encore définie sur ce gestionnaire de files d'attente
- Un message arrive dans un gestionnaire de files d'attente, mais la file d'attente à laquelle il est destiné ne peut pas le recevoir pour la raison suivante:
  - La file d'attente est saturée
  - Les demandes d'insertion sont interdites
  - Le noeud émetteur n'a pas le droit d'insérer des messages dans la file d'attente

Les applications peuvent également placer des messages dans la file d'attente des messages non livrés.

Les messages de rapport sont traités de la même manière que les messages ordinaires ; si le message de rapport ne peut pas être distribué à sa file d'attente de destination (généralement la file d'attente spécifiée par la zone *ReplyToQ* dans le descripteur de message du message d'origine), le message de rapport est placé dans la file d'attente de rebut (message non distribué).

**Remarque :** Les messages dont l'heure d'expiration a été dépassée (voir [MQMD-Zone d'expiration](#)) ne sont **pas** transférées à cette file d'attente lorsqu'elles sont supprimées. Toutefois, un message de rapport d'expiration (MQRO\_EXPIRATION) est toujours généré et envoyé à la file d'attente *ReplyToQ*, si l'application émettrice le demande.

Les messages ne sont pas insérés dans la file d'attente des messages non livrés lorsque l'application qui a émis la demande d'insertion a été informée de manière synchrone du problème à l'aide du code anomalie renvoyé par l'appel MQPUT ou MQPUT1 (par exemple, un message inséré dans une file d'attente locale pour laquelle les demandes d'insertion sont interdites).

Les messages de la file d'attente de messages non livrés ont parfois des données de message d'application préfixées avec une structure MQDLH. Cette structure contient des informations supplémentaires qui indiquent la raison pour laquelle le message a été placé dans la file d'attente des messages non livrés. Pour plus d'informations sur cette structure, voir [«MQDLH-En-tête de rebut»](#), à la page 364 .

Cette file d'attente doit être une file d'attente locale, avec l'attribut **Usage** MQUS\_NORMAL.

Si un gestionnaire de files d'attente ne prend pas en charge une file d'attente de rebut (message non distribué) ou si aucune file d'attente n'a été définie, le nom est à blanc. Tous les gestionnaires de files d'attente IBM MQ prennent en charge une file d'attente de rebut (message non distribué), mais elle n'est pas définie par défaut.

Si la file d'attente de rebut (messages non livrés) n'est pas définie, saturée ou inutilisable pour une autre raison, un message qui lui aurait été transféré par un agent de canal de transmission est conservé dans la file d'attente de transmission.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_DEAD\_LETTER\_Q\_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_Q\_NAME\_LENGTH.

### ***DefClusterXmitQueueType (MQLONG)***

L'attribut `DefClusterXmitQueue` contrôle la file d'attente de transmission sélectionnée par défaut par les canaux émetteurs de cluster à partir de laquelle les messages doivent être envoyés aux canaux récepteurs de cluster.

Les valeurs de `DefClusterXmitQueueType` sont `MQCLXQ_Q_Q` ou `MQCLXQ_CANAL`.

#### **MQCLXQ\_SCTQ**

Tous les canaux émetteurs de cluster envoient des messages à partir de `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. L'ID `correlID` de messages placés dans la file d'attente de transmission identifie le canal émetteur de cluster auquel le message est destiné.

SCTQ est défini lorsqu'un gestionnaire de files d'attente est défini.

#### **MQCLXQ\_CHANNEL**

Chaque canal émetteur de cluster envoie des messages à partir d'une file d'attente de transmission différente. Chaque file d'attente de transmission est créée en tant que file d'attente dynamique permanente à partir du modèle de file d'attente `SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE`.

Si l'attribut de gestionnaire de files d'attente `DefClusterXmitQueueType` est défini sur `CHANNEL`, La configuration par défaut est remplacée par une configuration dans laquelle les canaux émetteurs de cluster sont associés à des files d'attente de transmission de cluster individuelles. Les files d'attente de transmission sont des files d'attente dynamiques permanentes créées à partir de la file d'attente modèle `SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE`. Chacune d'elles est associée à un canal émetteur de cluster. Etant donné qu'un canal émetteur de cluster sert une file d'attente de transmission de cluster, cette dernière contient les messages relatifs à un seul gestionnaire de files d'attente d'un seul cluster. Vous pouvez configurer les clusters afin que chaque gestionnaire de files d'attente d'un cluster contienne une seule file d'attente de cluster. Dans le cas, le trafic de messages entre un gestionnaire de files d'attente et chaque file d'attente de cluster est transféré séparément des files d'attente de messages à d'autres files d'attente.

Pour interroger la valeur, appelez `MQINQ` ou envoyez une commande `Inquire Queue Manager` (`MQCMD_INQUIRE_Q_MGR`) PCF, en définissant le sélecteur `MQIA_DEF_CLUSTER_XMIT_Q_TYPE`. Pour modifier la valeur, envoyez une commande `PCF MQCMD_CHANGE_Q_MGR` (Modifier un gestionnaire de files d'attente), en définissant le sélecteur `MQIA_DEF_CLUSTER_XMIT_Q_TYPE`.

#### **Référence associée**

[Modifier un gestionnaire de files d'attente](#)

[Consulter les gestionnaires de files d'attente](#)

«MQINQ-Attributs d'objet Inquire», à la page 737

L'appel `MQINQ` renvoie un tableau d'entiers et un ensemble de chaînes de caractères contenant les attributs d'un objet.

### ***DefXmitQName (MQCHAR48)***

Il s'agit du nom de la file d'attente de transmission utilisée pour la transmission des messages aux gestionnaires de files d'attente éloignées, s'il n'existe aucune autre indication de la file d'attente de transmission à utiliser.

S'il n'existe pas de file d'attente de transmission par défaut, le nom est entièrement vide. La valeur initiale de cet attribut est vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur `MQCA_DEF_XMIT_Q_NAME` avec l'appel `MQINQ`. La longueur de cet attribut est donnée par `MQ_Q_NAME_LENGTH`.

### ***DistLists (MQLONG)***

Indique si le gestionnaire de files d'attente local prend en charge les listes de distribution dans les appels `MQPUT` et `MQPUT1`. Ses valeurs sont l'une des suivantes :

#### **MQDL\_SUPPORTED**

Listes de distribution prises en charge.

## **MQDL\_NOT\_SUPPORTED**

Les listes de distribution ne sont pas prises en charge.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_DIST\_LISTS avec l'appel MQINQ.

## **DNSGroup (MQCHAR18)**

Ce paramètre n'est plus utilisé. Voir [Modifications apportées à IBM MQ 8.0](#).

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_DNS\_GROUP avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_DNS\_GROUP\_NAME\_LENGTH.

## **DNSWLM (MQLONG)**

Ce paramètre n'est plus utilisé. Voir [Modifications apportées à IBM MQ 8.0](#).

La valeur est l'une des suivantes :

### **MQDNSWLM\_OUI**

Cette valeur est visible sur un gestionnaire de files d'attente migré à partir d'une édition antérieure. La valeur est ignorée.

### **MQDNSWLM\_NO**

Il s'agit de la seule valeur prise en charge par le gestionnaire de files d'attente.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_DNS\_WLM avec l'appel MQINQ.


## **ExpiryInterval (MQLONG)**

Indique la fréquence à laquelle le gestionnaire de files d'attente analyse les files d'attente à la recherche de messages arrivés à expiration. Il s'agit soit d'un intervalle de temps en secondes compris entre 1 et 99 999 999, soit de la valeur spéciale suivante:

### **MQEXPI\_OFF**

Le gestionnaire de files d'attente n'analyse pas les files d'attente à la recherche de messages arrivés à expiration.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_EXPIRY\_INTERVAL avec l'appel MQINQ.

 Cet attribut est pris en charge uniquement sous z/OS.

## **IGQPutAuthority (MQLONG)**

Cet attribut s'applique uniquement si le gestionnaire de files d'attente local est membre d'un groupe de partage de files d'attente. Il indique le type de vérification des droits d'accès effectuée lorsque l'agent de mise en file d'attente intra-groupe local (agent IGQ) supprime un message de la file d'attente de transmission partagée et place le message dans une file d'attente locale. La valeur est l'une des suivantes :

### **MQIGQPA\_PAR DEFALT**

L'identificateur utilisateur vérifié pour l'autorisation est la valeur de la zone *UserIdentifier* dans le MQMD *distinct* associé au message lorsque le message se trouve dans la file d'attente de transmission partagée. Il s'agit de l'ID utilisateur du programme qui a placé le message dans la file d'attente de transmission partagée et qui est généralement identique à l'ID utilisateur sous lequel le gestionnaire de files d'attente éloignées s'exécute.

Si le profil RESLEVEL indique que plusieurs identificateurs d'utilisateur doivent être vérifiés, l'identificateur d'utilisateur de l'agent IGQ local (*IGQUserId*) est également vérifié.



## **MQIGQA\_CONTEXT**

L'identificateur utilisateur vérifié pour l'autorisation est la valeur de la zone *UserIdentifier* dans le MQMD *distinct* associé au message lorsque le message se trouve dans la file d'attente de transmission partagée. Il s'agit de l'ID utilisateur du programme qui a placé le message dans la file d'attente de transmission partagée et qui est généralement identique à l'ID utilisateur sous lequel le gestionnaire de files d'attente éloignées s'exécute.

Si le profil RESLEVEL indique que plusieurs identificateurs d'utilisateur doivent être vérifiés, l'identificateur d'utilisateur de l'agent IGQ local (*IGQUserId*) et la valeur de la zone *UserIdentifier* dans le MQMD *intégré* sont également vérifiés. Ce dernier identifiant d'utilisateur est habituellement l'identifiant d'utilisateur de l'application à l'origine du message.

## **MQIGQA\_ONLY\_IGQ**

L'ID utilisateur vérifié pour l'autorisation est l'ID utilisateur de l'agent IGQ local (*IGQUserId*).


Si le profil RESLEVEL indique que plusieurs ID utilisateur doivent être vérifiés, cet ID utilisateur est utilisé pour toutes les vérifications.

## **MQIGQA\_ALTERNATE\_OR\_IGQ**

L'ID utilisateur vérifié pour l'autorisation est l'ID utilisateur de l'agent IGQ local (*IGQUserId*).

Si le profil RESLEVEL indique que plusieurs ID utilisateur doivent être vérifiés, la valeur de la zone *UserIdentifier* dans le MQMD *imbriqué* est également vérifiée. Il s'agit généralement de l'ID utilisateur de l'application à l'origine du message.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_IGQ\_PUT\_AUTHORITY avec l'appel MQINQ.


 Cet attribut est pris en charge uniquement sous z/OS.

## **IGQUserId (MQLONG)**

Cet attribut est applicable uniquement si le gestionnaire de files d'attente local est membre d'un groupe de partage de files d'attente. Il indique l'ID utilisateur associé à l'agent de mise en file d'attente intra-groupe local (agent de mise en file d'attente intra-groupe). Il s'agit de l'un des ID utilisateur dont l'autorisation peut être vérifiée lorsque l'agent de mise en file d'attente intra-groupe insère des messages dans des files d'attente locales. Les identificateurs d'utilisateur réels vérifiés dépendent de la définition de l'attribut **IGQPutAuthority** et des options de sécurité externes.

Si *IGQUserId* est vide, aucun identificateur d'utilisateur n'est associé à l'agent IGQ et la vérification d'autorisation correspondante n'est pas effectuée (bien que d'autres identificateurs d'utilisateur puissent encore être vérifiés pour l'autorisation).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_IGQ\_USER\_ID avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_USER\_ID\_LENGTH.

 Cet attribut est pris en charge uniquement sous z/OS.

## **InhibitEvent (MQLONG)**

Cette option contrôle si des événements d'inhibition (Inhibition Get et Inhibition Put) sont générés. La valeur est l'une des suivantes :

### **MQEVR\_DISABLED**

Génération de rapports d'événements désactivée.

### **MQEVR\_ENABLED**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_INHIBIT\_EVENT avec l'appel MQINQ.

Sous z/OS, vous ne pouvez pas utiliser l'appel MQINQ pour déterminer la valeur de cet attribut.

## ***IntraGroupqueuing (MQLONG)***

Cet attribut s'applique uniquement si le gestionnaire de files d'attente local est membre d'un groupe de partage de files d'attente. Indique si la mise en file d'attente intra-groupe est activée pour le groupe de partage de files d'attente. La valeur est l'une des suivantes :

### **MQIGQ\_DISABLED**

Tous les messages destinés aux autres gestionnaires de files d'attente du groupe de partage de files d'attente sont transmis à l'aide de canaux conventionnels.

### **MQIGQ\_ENABLED**

Les messages destinés aux autres gestionnaires de files d'attente du groupe de partage de files d'attente sont transmis à l'aide de la file d'attente de transmission partagée si la condition suivante est remplie:


- La longueur des données de message et de l'en-tête de transmission ne dépasse pas 63 Ko (64 512 octets).

Il est recommandé d'allouer un peu plus d'espace que la taille de MQXQH pour l'en-tête de transmission ; la constante MQ\_MSG\_HEADER\_LENGTH est fournie à cet effet.

Si cette condition n'est pas remplie, le message est transmis à l'aide de canaux classiques.

**Remarque :** Lorsque la mise en file d'attente intra-groupe est activée, l'ordre des messages transmis à l'aide de la file d'attente de transmission partagée n'est pas conservé par rapport à ceux transmis à l'aide des canaux classiques.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_INTRA\_GROUP\_queuing avec l'appel MQINQ.

 Cet attribut est pris en charge uniquement sous z/OS.

## ***IPAddressVersion (MQLONG)***

Indique quelle version d'adresse IP, IPv4 ou IPv6, est utilisée.

Cet attribut est pertinent uniquement pour les systèmes qui exécutent à la fois IPv4 et IPv6 et affecte uniquement les canaux définis comme ayant un *TransportType* de MQXPY\_TCP lorsque l'une des conditions suivantes est vérifiée:

- Le *ConnectionName* du canal est un nom d'hôte qui se résout en une adresse IPv4 et IPv6 et son paramètre **LocalAddress** n'est pas spécifié.
- Les noms d'hôte *ConnectionName* et *LocalAddress* du canal sont tous deux des noms d'hôte qui se résolvent à la fois en adresses IPv4 et IPv6 .

La valeur peut être l'une des valeurs suivantes :

### **MQIPADDR\_IPV4**

IPv4 est utilisée.

### **MQIPADDR\_IPV6**

IPv6 est utilisée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_IP\_ADDRESS\_VERSION avec l'appel MQINQ.

## ***ListenerTimer (MQLONG)***

Il s'agit de l'intervalle (en secondes) entre les tentatives de redémarrage du programme d'écoute par IBM MQ en cas d'échec APPC ou TCP/IP. La valeur doit être comprise entre 5 et 9999, avec une valeur par défaut de 60.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_LISTENER\_TIMER avec l'appel MQINQ.

### **LocalEvent (MQLONG)**

Cette option contrôle si des événements d'erreur locaux sont générés. La valeur est l'une des suivantes :

#### **MQEVN\_DISABLED**

Génération de rapports d'événements désactivée.

#### **MQEVN\_ENABLED**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_LOCAL\_EVENT avec l'appel MQINQ.

Sous z/OS, vous ne pouvez pas utiliser l'appel MQINQ pour déterminer la valeur de cet attribut.

### **LoggerEvent (MQLONG)**

Cette option contrôle si les événements du journal de reprise sont générés. La valeur est l'une des suivantes :

#### **MQEVN\_DISABLED**

Génération de rapports d'événements désactivée.

#### **MQEVN\_ENABLED**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_LOGGER\_EVENT avec l'appel MQINQ.

 Cet attribut est pris en charge uniquement sous [Multiplateformes](#).

### **LUGroupName (MQCHAR8)**

Il s'agit du nom d'unité logique générique du programme d'écoute d'unité logique 6.2 qui gère les transmissions entrantes pour le groupe de partage de files d'attente. Si vous laissez ce nom vide, vous ne pouvez pas utiliser ce programme d'écoute.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_LU\_GROUP\_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_LU\_NAME\_LENGTH.

### **Nom d'unité logique (MQCHAR8)**

Il s'agit du nom de l'unité logique à utiliser pour les transmissions LU 6.2 sortantes. Définissez cette valeur sur la même unité logique que celle utilisée par le programme d'écoute pour les transmissions entrantes. Si vous laissez ce nom vide, l'unité logique par défaut APPC/MVS est utilisée ; il s'agit d'une variable. Par conséquent, définissez toujours le nom d'unité logique si vous utilisez LU6.2.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_LU\_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_LU\_NAME\_LENGTH.

### **LU62ARMSuffix (MQCHAR2)**

Il s'agit du suffixe de SYS1.PARMLIB APPCPMxx, qui désigne le LUADD pour cet initiateur de canal. La commande z/OS SET APPC=xx est émise lorsque ARM redémarre l'initiateur de canal. Si vous laissez ce nom à blanc, aucun SET APPC=xx n'est émis.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_LU62\_ARM\_SUFFIX avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_ARM\_SUFFIX\_LENGTH.

## **LU62Channels (MQLONG)**

Il s'agit du nombre maximal de canaux pouvant être en cours, ou de clients pouvant être connectés, qui utilisent le protocole de transmission LU 6.2 .

La valeur doit être comprise entre 0 et 9999, avec une valeur par défaut de 200. Si vous définissez cette valeur sur zéro, le protocole de transmission LU 6.2 n'est pas utilisé.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_LU62\_CHANNELS avec l'appel MQINQ.

## **MaxActiveactifs (MQLONG)**

Cet attribut correspond au nombre maximal de canaux pouvant être *actifs* à tout moment.

La valeur par défaut est celle spécifiée pour l'attribut MaxChannels.

Pour z/OS, la valeur doit être comprise entre 1 et 9 999.

Pour toutes les autres plateformes, la valeur par défaut est 999 999 999, ce qui signifie que le nombre de canaux actifs est illimité, ou qu'il peut être défini sur un nombre réel pour imposer une limite.

**MQ Appliance** Vous ne devez pas modifier la valeur **MaxActiveChannels** sur IBM MQ Appliance. Si vous souhaitez limiter le nombre maximal de canaux client, utilisez les attributs MAXINST et MAXINSTC par canal sur les définitions de canal SVRCONN pour définir des limites pour chaque canal SVRCONN. Voir [Configuration du gestionnaire de files d'attente sur le dispositif IBM MQ Appliance](#) dans la documentation IBM MQ Appliance .

Le paramètre **MaxActiveChannels** est un attribut de gestionnaire de files d'attente sous z/OS uniquement. Sur les autres plateformes, **MaxActiveChannels** est un attribut du fichier `qm.ini` . Pour plus d'informations sur la définition de l'attribut **MaxActiveChannels** sur d'autres plateformes, voir [Sections du fichier de configuration pour la mise en file d'attente répartie](#) .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_ACTIVE\_CHANNELS avec l'appel MQINQ .

### **Concepts associés**

[Etats des canaux](#)

## **MaxChannels (MQLONG)**

Cet attribut correspond au nombre maximal de canaux pouvant être *en cours* (y compris les canaux de connexion serveur avec des clients connectés).

Pour z/OS, la valeur doit être comprise entre 1 et 9 999, avec une valeur par défaut de 200.

**MQ Appliance** Pour IBM MQ Appliance, la valeur par défaut est 999 999 999 et ne doit pas être modifiée. Si vous souhaitez limiter le nombre maximal de canaux client, utilisez les attributs MAXINST et MAXINSTC par canal sur les définitions de canal SVRCONN pour définir des limites pour chaque canal SVRCONN. Voir [Configuration du gestionnaire de files d'attente sur le dispositif IBM MQ Appliance](#) dans la documentation IBM MQ Appliance .

Un système qui est occupé à servir des connexions à partir du réseau peut avoir besoin d'un nombre supérieur à la valeur par défaut. Déterminez la valeur correcte pour votre environnement, idéalement en observant le comportement de votre système lors des tests.

Pour toutes les autres plateformes, la valeur par défaut est 100. Vous pouvez définir **MaxChannels** sur une valeur différente pour limiter le nombre maximal de canaux en cours si nécessaire.

Le paramètre **MaxChannels** est un attribut de gestionnaire de files d'attente sous z/OS uniquement. Sur les autres plateformes, **MaxChannels** est un attribut du fichier `qm.ini` . Pour plus d'informations sur la définition de l'attribut **MaxChannels** sur d'autres plateformes, voir [Sections du fichier de configuration pour la mise en file d'attente répartie](#) .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_MAX\_CHANNELS avec l'appel MQINQ.

### Concepts associés

[Etats des canaux](#)

### **MaxHandles (MQLONG)**

Il s'agit du nombre maximal de descripteurs ouverts pouvant être utilisés simultanément par une tâche. Chaque appel MQOPEN réussi pour une file d'attente unique (ou pour un objet qui n'est pas une file d'attente) utilise un descripteur. Ce descripteur devient disponible pour être réutilisé lorsque l'objet est fermé. Toutefois, lorsqu'une liste de distribution est ouverte, chaque file d'attente de la liste de distribution se voit attribuer un descripteur distinct, de sorte que l'appel MQOPEN utilise autant de descripteurs que de files d'attente de la liste de distribution. Cela doit être pris en compte lors de la décision d'une valeur appropriée pour *MaxHandles*.

L'appel MQPUT1 effectue un appel MQOPEN dans le cadre de son traitement ; par conséquent, MQPUT1 utilise autant de descripteurs que le ferait MQOPEN, mais les descripteurs ne sont utilisés que pendant la durée de l'appel MQPUT1 lui-même.

Sous z/OS, *task* signifie une tâche CICS, une tâche MVS ou une région dépendante IMS.

La valeur est comprise entre 1 et 999 999 999. La valeur par défaut est déterminée par l'environnement:

- Sous z/OS, la valeur par défaut est 100.
- Dans tous les autres environnements, la valeur par défaut est 256.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_MAX\_POIGNÉES avec l'appel MQINQ.

### **Longueur MaxMsg(MQLONG)**

Il s'agit de la longueur du message *physique* le plus long que le gestionnaire de files d'attente peut traiter. Toutefois, comme l'attribut de gestionnaire de files d'attente **MaxMsgLength** peut être défini indépendamment de l'attribut de file d'attente **MaxMsgLength**, le message physique le plus long pouvant être placé dans une file d'attente est la valeur la moins élevée de ces deux valeurs.

Si le gestionnaire de files d'attente prend en charge la segmentation, une application peut insérer un message logique plus long que le moindre des deux attributs **MaxMsgLength**, mais uniquement si l'application spécifie l'indicateur MQMF\_SEGMENTATION\_ALLOWED dans MQMD. Si cet indicateur est spécifié, la limite supérieure de la longueur d'un message logique est de 999 999 999 999 octets, mais les contraintes de ressources imposées par le système d'exploitation ou par l'environnement dans lequel l'application est exécutée entraînent une limite inférieure.

La limite inférieure de l'attribut **MaxMsgLength** est de 32 ko (32 768 octets). La limite supérieure est de 100 Mo (104 857 600 octets).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_MAX\_MSG\_LENGTH avec l'appel MQINQ.

### **MaxPriority (MQLONG)**

Il s'agit de la priorité de message maximale prise en charge par le gestionnaire de files d'attente. Les priorités sont comprises entre zéro (niveau le plus bas) et *MaxPriority* (niveau le plus élevé).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_MAX\_PRIORITY avec l'appel MQINQ.

### **MaxPropertiesLongueur (MQLONG)**

Permet de contrôler la taille des propriétés pouvant être associées à un message. Cela inclut à la fois le nom de la propriété en octets et la taille de la valeur de la propriété également en octets.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_MAX\_PROPERTIES\_LENGTH avec l'appel MQINQ.

## Multi **Taille MaxQFile(MQLONG)**

Taille maximale, en mégaoctets, qu'un fichier de file d'attente peut atteindre.

Tableau 560. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Il est possible qu'un fichier de file d'attente dépasse la taille maximale, s'il est configuré avec une valeur inférieure à la taille du fichier de file d'attente en cours. Si cela se produit, le fichier de file d'attente n'accepte plus les nouveaux messages, mais autorise l'utilisation des messages existants. Lorsque la taille du fichier de file d'attente est inférieure à la valeur configurée, les nouveaux messages peuvent être placés dans la file d'attente.

**Remarque :** Cette figure peut différer de la valeur de l'attribut configuré sur la file d'attente, car en interne, le gestionnaire de files d'attente peut avoir besoin d'utiliser une taille de bloc plus importante pour atteindre la taille choisie. Voir [Modification des fichiers de file d'attente IBM MQ](#) pour plus d'informations sur la modification de la taille des fichiers de file d'attente et de la taille et de la granularité des blocs.

Lorsque la granularité doit être modifiée car cet attribut a été augmenté, le message d'avertissement AMQ7493W *Granularity changed* est consigné dans les journaux AMQERR. Cela vous indique que vous devez planifier le vidage de la file d'attente afin que IBM MQ adopte la nouvelle granularité.

La valeur maximale de cet attribut est 267,386,880 Mo et la valeur par défaut, et la valeur migrée, est 2,088,960 Mo, ce qui correspond à la valeur maximale actuelle pour une file d'attente avec une granularité égale à 512.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_MAX\_Q\_FILE\_SIZE avec l'appel MQINQ.

## **Messages MaxUncommitted(MQLONG)**

Il s'agit du nombre maximal de messages non validés pouvant exister dans une unité de travail. Le nombre de messages non validés correspond à la somme des éléments suivants depuis le début de l'unité de travail en cours:

- Messages insérés par l'application avec l'option MQPMO\_SYNCPOINT
- Messages extraits par l'application avec l'option MQGMO\_SYNCPOINT
- Messages de déclenchement et messages de rapport COA générés par le gestionnaire de files d'attente pour les messages insérés avec l'option MQPMO\_SYNCPOINT
- Messages de rapport COD générés par le gestionnaire de files d'attente pour les messages extraits à l'aide de l'option MQGMO\_SYNCPOINT

Les messages suivants ne sont pas comptabilisés comme non validés:

- Messages insérés ou extraits par l'application en dehors d'une unité de travail
- Messages de déclenchement ou messages de rapport COA/COD générés par le gestionnaire de files d'attente suite à l'insertion ou à l'extraction de messages en dehors d'une unité d'oeuvre
- Messages de rapport d'expiration générés par le gestionnaire de files d'attente (même si l'appel à l'origine du message de rapport d'expiration a spécifié MQGMO\_SYNCPOINT)
- Messages d'événement générés par le gestionnaire de files d'attente (même si l'appel à l'origine du message d'événement a spécifié MQPMO\_SYNCPOINT ou MQGMO\_SYNCPOINT)

### **Remarque :**

1. Les messages de rapport d'exception sont générés par l'agent MCA (Message Channel Agent) ou par l'application et sont traités de la même manière que les messages ordinaires insérés ou extraits par l'application.

2. Lorsqu'un message ou un segment est inséré avec l'option MQPMO\_SYNCPOINT, le nombre de messages non validés est incrémenté de un, quel que soit le nombre de messages physiques résultant de l'insertion. (Plusieurs messages physiques peuvent être émis si le gestionnaire de files d'attente doit subdiviser le message ou le segment.)
3. Lorsqu'une liste de distribution est placée avec l'option MQPMO\_SYNCPOINT, le nombre de messages non validés est incrémenté d'un *pour chaque message physique généré*. Elle peut être aussi petite que 1 ou aussi grande que le nombre de destinations dans la liste de distribution.

La limite inférieure de cet attribut est 1 ; la limite supérieure est 999 999 999. La valeur par défaut est 10 000.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_MAX\_UNCOMMITTED\_MSGS avec l'appel MQINQ.

### **MQIAccounting (MQLONG)**

Permet de contrôler la collecte des informations de comptabilité pour les données MQI.

La valeur est l'une des suivantes :

#### **MQMON\_ON**

Collectez les données de comptabilité d'API.

#### **MQMON\_OFF**

Ne collectez pas de données de comptabilité d'API. Il s'agit de la valeur par défaut.

Si vous définissez l'attribut de gestionnaire de files d'attente ACCTCONO sur ENABLED, cette valeur peut être remplacée pour des connexions individuelles à l'aide de la zone Options de la structure MQCNO. Les modifications apportées à cette valeur ne sont effectives que pour les connexions au gestionnaire de files d'attente qui se produisent après la modification de l'attribut.

Cet attribut est pris en charge uniquement sous Multiplateformes.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_ACCOUNTING\_MQI avec l'appel MQINQ.

### **MQIStatistics (MQLONG)**

Cette option contrôle la collecte des informations de surveillance des statistiques pour le gestionnaire de files d'attente.

La valeur est l'une des suivantes :

#### **MQMON\_ON**

Collecte des statistiques MQI.

#### **MQMON\_OFF**

Ne collectez pas de statistiques MQI. Il s'agit de la valeur par défaut.

Cet attribut est pris en charge uniquement sous Multiplateformes.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_STATISTICS\_MQI avec l'appel MQINQ.

### **MsgMarkBrowseInterval (MQLONG)**

Intervalle de temps en millisecondes après lequel le gestionnaire de files d'attente peut supprimer automatiquement la marque des messages de navigation.

Il s'agit d'un intervalle de temps (en millisecondes) après lequel le gestionnaire de files d'attente peut supprimer automatiquement la marque des messages de navigation.

Cet attribut décrit l'intervalle de temps pendant lequel les messages qui ont été marqués comme parcourus par un appel à MQGET, à l'aide de l'option d'obtention de message MQGMO\_MARK\_BROWSE\_CO\_OP, doivent rester marqués comme parcourus.

Le gestionnaire de files d'attente peut automatiquement annuler le marquage des messages consultés qui ont été marqués comme ayant été consultés pour l'ensemble de descripteurs coopérant lorsqu'ils ont été marqués pour une durée supérieure à cet intervalle approximatif.

Cela n'affecte pas l'état d'un message marqué comme Parcourir, qui a été obtenu par un appel à MQGET, à l'aide de l'option d'obtention de message MQGMO\_MARK\_BROWSE\_HANDLE.

La valeur maximale est 999 999 999 et la valeur par défaut est 5000. La valeur spéciale -1 pour *MsgMarkBrowseInterval* représente un intervalle de temps illimité.



**Avertissement :** Cette valeur ne doit pas être inférieure à la valeur par défaut de 5000.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_MSG\_MARK\_BROWSE\_INTERVAL avec l'appel MQINQ.

### ***OutboundPortMax (MQLONG)***

Il s'agit du numéro de port le plus élevé dans la plage, définie par OutboundPortMin et OutboundPortMax, des numéros de port à utiliser pour lier les canaux sortants.

La valeur est un entier compris entre 0 et 65535 et doit être supérieure ou égale à la valeur minimale de OutboundPort. La valeur par défaut est 0.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_OUTBOUND\_PORT\_MAX avec l'appel MQINQ.

### ***OutboundPortMin (MQLONG)***

Il s'agit du numéro de port le plus bas de la plage, définie par OutboundPortMin et OutboundPortMax, des numéros de port à utiliser pour lier les canaux sortants.

La valeur est un entier compris entre 0 et 65535 et doit être inférieure ou égale à la valeur maximale de OutboundPort. La valeur par défaut est 0.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_OUTBOUND\_PORT\_MIN avec l'appel MQINQ.

### ***PerformanceEvent (MQLONG)***

Permet de contrôler si des événements liés aux performances sont générés. Ses valeurs sont l'une des suivantes :

#### **MQEVR\_DISABLED**

Génération de rapports d'événements désactivée.

#### **MQEVR\_ENABLED**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_PERFORMANCE\_EVENT avec l'appel MQINQ.

### ***Plateforme (MQLONG)***

Indique le système d'exploitation sur lequel le gestionnaire de files d'attente s'exécute:

#### **MQPL\_AIX**

AIX (même valeur que MQPL\_UNIX).

#### **MQPL\_DISPOSITIF**

IBM MQ Appliance

#### **MQPL\_MVS**

z/OS (même valeur que MQPL\_ZOS).



**MQPL\_OS390**

z/OS (même valeur que MQPL\_ZOS).

**MQPL\_OS400**

IBM i.

**MQPL\_UNIX**

UNIX.

**MQPL\_WINDOWS\_NT**

Windows cible.

**MQPL\_ZOS**

z/OS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_PLATFORM avec l'appel MQINQ.

***PubSubNPInputMsg (MQLONG)***

Indique s'il faut supprimer ou conserver un message d'entrée non distribué.

La valeur est l'une des suivantes :

**Carte MQUNDELIVERED\_DISCARD**

Les message en entrée non persistants peuvent être supprimés s'ils ne peuvent pas être traités.

Il s'agit de la valeur par défaut.

**MQUNDELIVERED\_KEEP**

Les message en entrée non persistants ne sont pas supprimés s'ils ne peuvent pas être traités.

Dans ce cas, l'interface de publication / abonnement mise en file d'attente continuera à relancer le processus à des intervalles appropriés et ne continuera pas à traiter les messages suivants.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_PUBSUB\_NP\_MSG avec l'appel MQINQ.

***PubSubPubSub (MQLONG)***

Contrôle le comportement des messages de réponse non distribués.

La valeur est l'une des suivantes :

**Mqundelivered\_normal**

Les réponses non persistantes qui ne peuvent pas être placées dans la file d'attente de réponses sont placées dans la file d'attente des messages non livrés. Si elles ne peuvent pas être placées dans la file d'attente des messages non livrés, elles sont supprimées.

**MQUNDELIVERED\_SAFE**

Les réponses non persistantes ne pouvant être placées dans la file de réponses le sont dans la file de rebut. Si la réponse ne peut pas être définie et ne peut pas être placée dans la file d'attente des messages non livrés, l'interface de publication / abonnement mise en file d'attente annule l'opération en cours, puis effectue une nouvelle tentative à des intervalles appropriés et ne poursuit pas le traitement des messages suivants.

**Carte MQUNDELIVERED\_DISCARD**

Les réponses non persistantes ne sont pas placées dans la file d'attente de réponses et sont supprimées.

Il s'agit de la valeur par défaut pour les nouveaux gestionnaires de files d'attente.

**MQUNDELIVERED\_KEEP**

Les réponses non persistantes ne sont pas placées dans la file d'attente des messages non livrés ni supprimées. A la place, l'interface de publication / abonnement mise en file d'attente renverra l'opération en cours, puis la relancera à des intervalles appropriés.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_PUBSUB\_NP\_RESP avec l'appel MQINQ.

## Valeur par défaut pour les gestionnaires de files d'attente migrés.

Si le gestionnaire de files d'attente a été migré depuis IBM MQ V6.0, la valeur initiale de cet attribut dépend des valeurs de *DiscardNonPersistentResponse* et *DLQNonPersistentResponse* avant la migration, comme indiqué dans le tableau suivant.

		Réponse DLQNonPersistent		
		Oui	Non	Non définie
DiscardNonPersistentResponse	Oui	Mqundelivered_normal	Carte MQUNDELIVERED_DISCARD	Mqundelivered_normal
	Non	MQUNDELIVERED_SAFE	MQUNDELIVERED_KEEP	MQUNDELIVERED_SAFE
	Non définie	Si SyncPointPersistent = No, MQUNDELIVERED_SAFE sinon MQUNDELIVERED_NORMAL	Si SyncPointPersistent = No, MQUNDELIVERED_KEEP sinon MQUNDELIVERED_DISCARD	Si SyncPointPersistent = No, MQUNDELIVERED_SAFE sinon MQUNDELIVERED_NORMAL

### **PubSubMaxMsgRetryCount (MQLONG)**

Nombre de nouvelles tentatives lors du traitement d'un message de commande ayant échoué sous le point de synchronisation.

La valeur est l'une des suivantes :

**0 - 999 999 999**

La valeur par défaut est 5.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_PUBSUB\_MAXMSG\_RETRY\_COUNT avec l'appel MQINQ.

### **PubSubSyncPoint (MQLONG)**

Indique si seuls les messages persistants ou tous les messages sont traités sous le point de synchronisation.

La valeur est l'une des suivantes :

#### **MQSYNCPOINT\_IFPER**

Cela permet à l'interface de publication / abonnement en file d'attente de recevoir des messages non persistants en dehors du point de synchronisation. S'il reçoit une publication hors du point de synchronisation, il la transmet aux abonnés qu'il connaît hors du point de synchronisation.

Il s'agit de la valeur par défaut.

#### **MQSYNCPOINT\_OUI**

Cela permet à l'interface de publication / abonnement en file d'attente de recevoir tous les messages sous le point de synchronisation.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_PUBSUB\_SYNC\_PT avec l'appel MQINQ.

### **Mode PubSub(MQLONG)**

Indique si le moteur de publication / abonnement et l'interface de publication / abonnement en file d'attente sont en cours d'exécution, ce qui permet aux applications de publier / abonnement à l'aide de l'interface de programmation d'application et des files d'attente surveillées par l'interface de publication / abonnement en file d'attente.

La valeur est l'une des suivantes :

#### **MQPSM\_COMPAT**

Le moteur publication/abonnement est en cours d'exécution. Il est donc possible de publier / s'abonner à l'aide de l'interface de programmation d'application. L'interface de publication / abonnement en file d'attente n'est pas en cours d'exécution. Par conséquent, aucun message inséré dans les files d'attente surveillées par l'interface de publication / abonnement en file d'attente n'est traité. Ce paramètre est utilisé à des fins de compatibilité avec WebSphere Message Broker V6 ou des versions antérieures utilisant ce gestionnaire de files d'attente, car il doit lire les mêmes files d'attente que celles à partir desquelles l'interface de publication / abonnement en file d'attente lit normalement.

## **MQPSM\_DISABLED**

Le moteur pub./abon. et l'interface pub./abon. en file d'attente sont inactifs. Il n'est donc pas possible de publier / s'abonner à l'aide de l'interface de programmation d'application. Les messages de publication / abonnement placés dans les files d'attente surveillées par l'interface de publication / abonnement en file d'attente ne sont pas traités.

## **MQPSM\_ENABLED**

Le moteur publication/abonnement et l'interface publication/abonnement sont en cours d'exécution. Il est donc possible de publier / abonner à l'aide de l'interface de programmation d'application et des files d'attente surveillées par l'interface de publication / abonnement en file d'attente. Il s'agit de la valeur par défaut initiale du gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_PUBSUB\_MODE avec l'appel MQINQ.

## **QMGrDesc (MQCHAR64)**

Utilisez cette zone pour un commentaire décrivant le gestionnaire de files d'attente. Le contenu de la zone n'est pas significatif pour le gestionnaire de files d'attente, mais ce dernier peut exiger que la zone ne contienne que des caractères pouvant être affichés. Elle ne peut pas contenir de caractères nuls ; si nécessaire, elle est remplie à droite avec des blancs. Dans une installation DBCS, cette zone peut contenir des caractères DBCS (avec une longueur de zone maximale de 64 octets).

**Remarque :** Si cette zone contient des caractères qui ne figurent pas dans le jeu de caractères du gestionnaire de files d'attente (tel que défini par l'attribut de gestionnaire de files d'attente **CodedCharSetId**), ces caractères peuvent être convertis de manière incorrecte si cette zone est envoyée à un autre gestionnaire de files d'attente.

- Sous z/OS, la valeur par défaut est le nom du produit et le numéro de version.
- Dans tous les autres environnements, la valeur par défaut est vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_Q\_MGR\_DESC avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_Q\_MGR\_DESC\_LENGTH.

## **QMGrIdentifieur (MQCHAR48)**

Il s'agit d'un nom unique généré en interne pour le gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_Q\_MGR\_IDENTIFIEUR avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_Q\_MGR\_IDENTIFIEUR\_LENGTH.

Cet attribut est pris en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows
-  z/OS

et les clients IBM MQ connectés à ces systèmes.

## **QMGrName (MQCHAR48)**

Il s'agit du nom du gestionnaire de files d'attente local, c'est-à-dire du nom du gestionnaire de files d'attente auquel l'application est connectée.

Les 12 premiers caractères du nom sont utilisés pour construire un identificateur de message unique (voir la zone [MQMD- MsgId](#)). Les gestionnaires de files d'attente qui peuvent communiquer entre eux doivent donc avoir des noms qui diffèrent dans les 12 premiers caractères, afin que les identificateurs de message soient uniques dans le réseau du gestionnaire de files d'attente.


Sous z/OS, le nom est identique au nom du sous-système, qui est limité à 4 caractères non blancs.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_Q\_MGR\_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_Q\_MGR\_NAME\_LENGTH.

### **QSGName (MQCHAR4)**

Il s'agit du nom du groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local. Si le gestionnaire de files d'attente local n'appartient pas à un groupe de partage de files d'attente, le nom est vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_QSG\_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_QSG\_NAME\_LENGTH.

 Cet attribut est pris en charge uniquement sous z/OS.

### **QueueAccounting (MQLONG)**

Permet de contrôler la collecte des informations de comptabilité pour les files d'attente.

La valeur est l'une des suivantes :

#### **MQMON\_AUCUN**

Ne collectez pas de données de comptabilité pour les files d'attente, quelle que soit la valeur de l'attribut de comptabilité de file d'attente ACCTQ. Il s'agit de la valeur par défaut.

#### **MQMON\_OFF**

Ne collectez pas de données comptables pour les files d'attente qui spécifient QMGR dans l'attribut de file d'attente ACCTQ.

#### **MQMON\_ON**

Collectez les données de comptabilité pour les files d'attente qui spécifient QMGR dans l'attribut de file d'attente ACCTQ.

Les modifications apportées à cette valeur ne sont effectives que pour les connexions au gestionnaire de files d'attente qui se produisent après la modification de l'attribut.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_ACCOUNTING\_Q avec l'appel MQINQ.

### **QueueMonitoring (MQLONG)**

Indique le paramètre par défaut pour la surveillance en ligne des files d'attente.

Si l'attribut de file d'attente **QueueMonitoring** est défini sur MQMON\_Q\_MGR, cet attribut spécifie la valeur utilisée par le canal. La valeur peut être :

#### **MQMON\_OFF**

La collecte des données de surveillance en ligne est désactivée. Il s'agit de la valeur par défaut initiale du gestionnaire de files d'attente.

#### **MQMON\_AUCUN**

La collecte des données de surveillance en ligne est désactivée pour les files d'attente, quelle que soit la valeur de leur attribut **QueueMonitoring**.

#### **MQMON\_FAIBLE**

La collecte de données de surveillance en ligne est activée, avec un faible taux de collecte de données.

#### **MQMON\_MEDIUM**

La collecte de données de surveillance en ligne est activée, avec un taux modéré de collecte de données.

#### **MQMON\_ELEVE**

La collecte de données de surveillance en ligne est activée, avec un taux élevé de collecte de données.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_MONITORING\_Q avec l'appel MQINQ.

### **QueueStatistics (MQLONG)**

Cette option contrôle la collecte des données statistiques pour les files d'attente.

Ses valeurs sont l'une des suivantes :

#### **MQMON\_AUCUN**

Ne collectez pas de statistiques de file d'attente pour les files d'attente, quel que soit le paramètre de l'attribut de file d'attente **QueueStatistics** . Il s'agit de la valeur par défaut.

#### **MQMON\_OFF**

Ne collectez pas de données statistiques pour les files d'attente qui spécifient le gestionnaire de files d'attente dans l'attribut de file d'attente **QueueStatistics** .

#### **MQMON\_ON**

Collectez des données statistiques pour les files d'attente qui spécifient le gestionnaire de files d'attente dans l'attribut de file d'attente **QueueStatistics** .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_STATISTICS\_Q avec l'appel MQINQ.

### ***ReceiveTimeout (MQLONG)***

Indique la durée pendant laquelle un canal TCP/IP attend de recevoir des données, y compris des signaux de présence, de son partenaire avant de revenir à l'état inactif. Elle s'applique uniquement aux canaux de message et non aux canaux MQI.

La signification exacte de ReceiveTimeout est modifiée par la valeur spécifiée dans le type ReceiveTimeout. Le type ReceiveTimeout peut être défini sur l'une des valeurs suivantes:

- MQRCVTIME\_EQUAL-cette valeur correspond au nombre de secondes d'attente du canal. Indiquez une valeur comprise entre 0 et 999999.
- MQRCVTIME\_ADD-cette valeur correspond au nombre de secondes à ajouter à l'élément HBINT négocié et détermine la durée d'attente d'un canal. Indiquez une valeur comprise entre 1 et 999999.
- MQRCVTIME\_MULTIPLIER-cette valeur est un multiplicateur à appliquer au HBINT négocié. Indiquez une valeur de 0 ou une valeur comprise entre 2 et 99.

La valeur par défaut est 0.

Définissez ReceiveTimeout sur MQRCVTIME\_MULTIPLIER ou MQRCVTIME\_EQUAL, et sur ReceiveTimeout sur 0, pour empêcher un canal d'expirer son attente de réception des données de son partenaire.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_RECEIVE\_TIMEOUT avec l'appel MQINQ.

### ***ReceiveTimeoutMin (MQLONG)***

Il s'agit de la durée minimale, en secondes, pendant laquelle un canal TCP/IP attend de recevoir des données, y compris des signaux de présence, de la part de son partenaire, avant de revenir à l'état inactif.

Elle s'applique uniquement aux canaux de transmission de messages et non aux canaux MQI. La valeur doit être comprise entre 0 et 999999, la valeur par défaut étant 0.

Si vous utilisez le type ReceiveTimeout pour indiquer que le temps d'attente du canal TCP/IP doit être calculé par rapport à la valeur négociée de HBINT et que la valeur résultante est inférieure à la valeur de ce paramètre, cette valeur est utilisée à la place.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_RECEIVE\_TIMEOUT\_MIN avec l'appel MQINQ.

### ***ReceiveTimeoutType (MQLONG)***

Il s'agit du qualificateur appliqué à ReceiveTimeout pour définir la durée pendant laquelle un canal TCP/IP attend la réception de données, y compris les signaux de présence, de la part de son partenaire, avant de

revenir à l'état inactif. Elle s'applique uniquement aux canaux de transmission de messages et non aux canaux MQI.

La valeur est l'une des suivantes :

#### **MQRCVTIME\_MULTIPLIER**

ReceiveTimeout est un multiplicateur à appliquer à la valeur HBINT négociée pour déterminer la durée d'attente d'un canal. Il s'agit de la valeur par défaut.

#### **MQRCVTIME\_ADD**

ReceiveTimeout est une valeur, en secondes, à ajouter à la valeur HBINT négociée pour déterminer la durée d'attente d'un canal.

#### **MQRCVTIME\_EQUAL**

ReceiveTimeout est une valeur, en secondes, que le canal attend.

Pour arrêter un dépassement du délai d'attente d'un canal pour recevoir des données de son partenaire, définissez le type ReceiveTimeout sur MQRCVTIME\_MULTIPLIER ou MQRCVTIME\_EQUAL et ReceiveTimeout sur 0.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_RECEIVE\_TIMEOUT\_TYPE avec l'appel MQINQ.

### ***RemoteEvent (MQLONG)***

Permet de contrôler si des événements d'erreur distants sont générés. Ses valeurs sont l'une des suivantes :

#### **MQEVN\_DISABLED**

Génération de rapports d'événements désactivée.

#### **MQEVN\_ENABLED**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_REMOTE\_EVENT avec l'appel MQINQ.

### ***RepositoryName (MQCHAR48)***

Il s'agit du nom d'un cluster pour lequel ce gestionnaire de files d'attente fournit un service de gestionnaire de référentiels. Si le gestionnaire de files d'attente fournit ce service pour plusieurs clusters, *RepositoryNameList* spécifie le nom d'un objet liste de noms qui identifie les clusters et *RepositoryName* est vide. Au moins l'un des éléments *RepositoryName* et *RepositoryNameList* doit être vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_REPOSITORY\_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_Q\_MGR\_NAME\_LENGTH.

### ***RepositoryNameList (MQCHAR48)***

Il s'agit du nom d'un objet liste de noms qui contient les noms des clusters pour lesquels ce gestionnaire de files d'attente fournit un service de gestionnaire de référentiels. Si le gestionnaire de files d'attente fournit ce service pour un seul cluster, l'objet liste de noms ne contient qu'un seul nom. Vous pouvez également utiliser *RepositoryName* pour spécifier le nom du cluster, auquel cas *RepositoryNameList* est vide. Au moins l'un des éléments *RepositoryName* et *RepositoryNameList* doit être vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_REPOSITORY\_NAMELIST avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_NAMELIST\_NAME\_LENGTH.

### ***ScyCase(MQCHAR8)***

Indique si le gestionnaire de files d'attente prend en charge les noms de profil de sécurité en casse mixte ou en majuscules uniquement.

La valeur est l'une des suivantes :


### **MQSCYC\_SUPÉRIEURE**

Les noms de profil de sécurité doivent être en majuscules.

### **MQSCYC\_MIXTE**

Les noms de profil de sécurité peuvent être en majuscules ou en casse mixte.

Les modifications apportées à cet attribut prennent effet lorsqu'une commande Refresh Security est exécutée avec *SecurityType* (MQSECTYPE\_CLASSES) spécifié.

 Cet attribut est pris en charge uniquement sous z/OS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_SECURITY\_CASE avec l'appel MQINQ.

### **Nom SharedQMgr(MQLONG)**

Indique si *ObjectQmgrName* doit être utilisé ou traité en tant que gestionnaire de files d'attente local sur un appel MQOPEN, pour une file d'attente partagée, lorsque *ObjectQmgrName* est celui d'un autre gestionnaire de files d'attente du groupe de partage de files d'attente.

La valeur peut être l'une des valeurs suivantes :

#### **MQSQQM\_USE**

*ObjectQmgrName* est utilisé et la file d'attente de transmission appropriée est ouverte.

#### **MQSQQM\_IGNORE**

Si la file d'attente cible est partagée et que *ObjectQmgrName* est celle d'un gestionnaire de files d'attente du même groupe de partage de files d'attente, l'ouverture est effectuée en local.

Cet attribut est valide uniquement sous z/OS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_SHARED\_Q\_Q\_MGR\_NAME avec l'appel MQINQ.

### **SPLCAP**

Indique si les fonctions de sécurité de Advanced Message Security sont disponibles pour un gestionnaire de files d'attente.

#### **MQCAP\_PRIS en charge**

Il s'agit de la valeur par défaut si le composant AMS est installé pour l'installation sous laquelle s'exécute le gestionnaire de files d'attente.

#### **MQCAP\_NON\_PRIS en charge**

### **SSLEvent (MQLONG)**

Indique si des événements TLS sont générés.

Ses valeurs sont l'une des suivantes :

#### **MQEVR\_ENABLED**

Générez des événements TLS, comme suit:

MQRC\_CHANNEL\_SSL\_ERROR

#### **MQEVR\_DISABLED**

Ne générez pas d'événements TLS ; il s'agit de la valeur par défaut.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_SSL\_EVENT avec l'appel MQINQ.

### **SSLFIPSRequired (MQLONG)**

**Remarque :** Sous AIX, Linux, and Windows, IBM MQ fournit la conformité à la norme FIPS 140-2 via le module cryptographique IBM Crypto for C (ICC) . Le certificat de ce module a été déplacé vers le statut Historique. Les clients doivent afficher le certificat IBM Crypto for C (ICC) et prendre connaissance des conseils fournis par le NIST. Un module FIPS 140-3 de remplacement est actuellement en cours et son statut peut être affiché en le recherchant dans la liste des modules NIST CMVP en cours de traitement.

Cela vous permet de spécifier que seuls les algorithmes certifiés FIPS doivent être utilisés si la cryptographie est exécutée dans IBM MQ, plutôt que dans le matériel de cryptographie. Si du matériel de cryptographie est configuré, les modules de cryptographie utilisés sont ceux fournis par le produit matériel ; ces modules peuvent ou non être certifiés FIPS à un niveau particulier en fonction du produit matériel utilisé.

La valeur est l'une des suivantes:

**MQSSL\_FIPS\_NO**

Utilisez tout CipherSpec pris en charge sur la plateforme utilisée. Cette valeur est la valeur par défaut.

**MQSSL\_FIPS\_YES**

Utilisez uniquement les algorithmes de cryptographie certifiés FIPS dans les CipherSpecs autorisés sur toutes les connexions TLS depuis et vers ce gestionnaire de files d'attente.

Ce paramètre est valide uniquement sur les plateformes z/OS, AIX, Linux, and Windows .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_SSL\_FIPS\_REQUIRED avec l'appel MQINQ.

**Tâches associées**

Comment indiquer que seuls les CipherSpecs certifiés FIPS sont utilisés lors de l'exécution sur MQI Client

**Référence associée**

FIPS (Federal Information Processing Standards) pour AIX, Linux, and Windows

**Nombre SSLKeyReset(MQLONG)**

Indique à quel moment les agents de canal de transmission de messages (MCA) TLS qui initient la communication réinitialisent la clé secrète utilisée pour le chiffrement sur le canal.

La valeur représente le nombre total d'octets non chiffrés qui sont envoyés et reçus sur le canal avant la renégociation de la clé secrète. Le nombre d'octets inclut les informations de contrôle envoyées par l'agent MCA.

La valeur est un nombre compris entre 0 et 999 999 999, avec une valeur par défaut de 0. Si vous spécifiez un nombre de réinitialisations de clé confidentielle TLS compris entre 1 octet et 32 Ko, les canaux TLS utiliseront un nombre de réinitialisations de clé confidentielle de 32 Ko. Cela permet d'éviter le coût de traitement d'un trop grand nombre de réinitialisations de clé qui se produiraient pour des valeurs de réinitialisation de clé secrète TLS de petite taille.

La clé secrète est renégociée lorsque le nombre total d'octets non chiffrés envoyés et reçus par l'agent MCA du canal initiateur dépasse la valeur spécifiée. Si les pulsations de canal sont activées, la clé secrète est renégociée avant que les données ne soient envoyées ou reçues à la suite d'une pulsation de canal, ou lorsque le nombre total d'octets non chiffrés dépasse la valeur spécifiée, selon la première de ces deux valeurs.

Le nombre d'octets envoyés et reçus pour la renégociation inclut les informations de contrôle envoyées et reçues par l'agent MCA de canal et est réinitialisé chaque fois qu'une renégociation se produit.

Utilisez la valeur 0 pour indiquer que les clés secrètes ne sont jamais renégociées.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_SSL\_RESET\_COUNT avec l'appel MQINQ.

**Événement StartStop(MQLONG)**

Cette option contrôle si les événements de démarrage et d'arrêt sont générés. La valeur est l'une des suivantes :

**MQEVR\_DISABLED**

Génération de rapports d'événements désactivée.

**MQEVR\_ENABLED**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir Surveillance des événements.



Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_START\_STOP\_EVENT avec l'appel MQINQ.

### ***StatisticsInterval (MQLONG)***

Indique la fréquence (en secondes) à laquelle les données de surveillance des statistiques doivent être écrites dans la file d'attente de surveillance.

La valeur est un entier compris entre 0 et 604800, avec une valeur par défaut de 1800 (30 minutes).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_STATISTICS\_INTERVAL avec l'appel MQINQ.

### ***SyncPoint (MQLONG)***

Indique si le gestionnaire de files d'attente local prend en charge les unités de travail et la synchronisation avec les appels MQGET, MQPUT et MQPUT1 .

#### **MQSP\_XX\_ENCODE\_CASE\_ONE disponible**

Unités de travail et point de synchronisation disponibles.

#### **MQSP\_NON DISPONIBLE**

Unités de travail et point de synchronisation non disponibles.

- Sous z/OS , cette valeur n'est jamais renvoyée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_SYNCPOINT avec l'appel MQINQ.

### ***TCPChannels (MQLONG)***

Il s'agit du nombre maximal de canaux pouvant être en cours, ou de clients pouvant être connectés, qui utilisent le protocole de transmission TCP/IP.

La valeur doit être comprise entre 0 et 9999, avec une valeur par défaut de 200. Si vous indiquez 0, TCP/IP n'est pas utilisé.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_TCP\_CHANNELS avec l'appel MQINQ.

### ***TCPKeepAlive (MQLONG)***

Indique si TCP KEEPALIVE doit être utilisé pour vérifier que l'autre extrémité de la connexion est toujours disponible. S'il n'est pas disponible, le canal est fermé.

La valeur est l'une des suivantes :

#### **MQTCPKEEP\_OUI**

Utilisez TCP KEEPALIVE comme indiqué dans le fichier de configuration de profil TCP. Si vous spécifiez l'attribut de canal KeepAliveInterval (KAINT), la valeur à laquelle il est défini est utilisée.

#### **MQTCPKEEP\_NO**

N'utilisez pas TCP KEEPALIVE. Il s'agit de la valeur par défaut.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_TCP\_KEEP\_ALIVE avec l'appel MQINQ.

### ***TCPName (MQCHAR8)***

Il s'agit du nom de la pile TCP/IP unique ou préférée qui sera utilisée, en fonction de la valeur de TCPStackType. Ce paramètre n'est applicable que dans les environnements à piles multiples CINET. La valeur par défaut est TCPIP.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_TCP\_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_TCP\_NAME\_LENGTH.

### ***TCPStackType (MQLONG)***

Indique si l'initiateur de canal peut utiliser uniquement la pile TCP/IP spécifiée dans TCPName ou peut éventuellement se lier à n'importe quelle pile TCP/IP sélectionnée. Ce paramètre n'est applicable que dans les environnements à piles multiples CINET.

La valeur est l'une des suivantes :

#### **MQTCPSTACK\_SINGLE**

L'initiateur de canal peut utiliser uniquement les espaces adresse TCP/IP nommés dans TCPName. Il s'agit de la valeur par défaut.

#### **MQTCPSTACK\_MULTIPLE**

L'initiateur de canal peut utiliser n'importe quel espace adresse TCP/IP disponible. La valeur par défaut est celle indiquée dans TCPName si aucune autre valeur n'est indiquée pour un canal ou un programme d'écoute.

Cet attribut est pris en charge sous z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_TCP\_STACK\_TYPE avec l'appel MQINQ.

### ***Enregistrement de TraceRoute(MQLONG)***

Permet de contrôler l'enregistrement des informations de routage de trace.

La valeur est l'une des suivantes :

#### **MQRECORDING\_DISABLED**

Aucun ajout aux messages de trace-route n'est autorisé.

#### **MQRECORDING\_Q**

Insertion de messages de trace-route dans une file d'attente nommée fixe.

#### **MSG\_ENREGISTREMENT\_MQG**

Placez les messages de trace-route dans une file d'attente déterminée à l'aide du message lui-même. Il s'agit de la valeur par défaut

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_TRACE\_ROUTE\_RECORDING avec l'appel MQINQ.

### ***TriggerInterval (MQLONG)***

Il s'agit d'un intervalle de temps (en millisecondes) utilisé pour limiter le nombre de messages de déclenchement. Cela n'est pertinent que si *TriggerType* est MQTT\_FIRST. Dans ce cas, les messages de déclenchement sont généralement générés uniquement lorsqu'un message approprié arrive dans la file d'attente et que la file d'attente était précédemment vide. Dans certaines circonstances, toutefois, un message de déclenchement supplémentaire peut être généré avec le déclenchement MQTT\_FIRST même si la file d'attente n'était pas vide. Ces messages de déclenchement supplémentaires ne sont pas générés plus souvent que toutes les *TriggerInterval* millisecondes.

Pour plus d'informations sur le déclenchement, voir [Déclenchement de canaux](#).

La valeur n'est pas inférieure à 0 ni supérieure à 999 999 999. La valeur par défaut est 999 999 999.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_TRIGGER\_INTERVAL avec l'appel MQINQ.

### ***TriggerInterval (MQLONG)***

Il s'agit d'un intervalle de temps (en millisecondes) utilisé pour limiter le nombre de messages de déclenchement. Cela n'est pertinent que si *TriggerType* est MQTT\_FIRST. Dans ce cas, les messages de déclenchement sont généralement générés uniquement lorsqu'un message approprié arrive dans la file d'attente et que la file d'attente était précédemment vide. Dans certaines circonstances, toutefois, un message de déclenchement supplémentaire peut être généré avec le déclenchement MQTT\_FIRST même

si la file d'attente n'était pas vide. Ces messages de déclenchement supplémentaires ne sont pas générés plus souvent que toutes les *TriggerInterval* millisecondes.

Pour plus d'informations sur le déclenchement, voir [Déclenchement de canaux](#).

La valeur n'est pas inférieure à 0 ni supérieure à 999 999 999. La valeur par défaut est 999 999 999.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_TRIGGER\_INTERVAL avec l'appel MQINQ.

### **Version (MQCFST)**

Il s'agit de la version du code IBM MQ en tant que VVRRMMFF, où:

VV-Version

RR-Libération

MM-Niveau de maintenance

FF-Niveau de correctif

### **XrCapability(MQLONG)**

Indique si les commandes MQ Telemetry sont prises en charge par le gestionnaire de files d'attente.

La valeur est l'une des suivantes :

#### **MQCAP\_PRIS en charge**

Le composant MQ Telemetry installé et les commandes de télémétrie sont prises en charge.

#### **MQCAP\_NON\_PRIS en charge**

Composant MQ Telemetry non installé.

Cet attribut est pris en charge uniquement sous [Multiplateformes](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_XR\_CAPABILITY avec l'appel MQINQ .

## **Attributs des files d'attente**


Il existe cinq types de définition de file d'attente. Certains attributs de file d'attente s'appliquent à tous les types de file d'attente ; d'autres s'appliquent uniquement à certains types de file d'attente.

## **Types de file d'attente**

Le gestionnaire de files d'attente prend en charge les types de définition de file d'attente suivants:

### **File d'attente locale**

Vous pouvez stocker des messages dans une file d'attente locale.


 Sous z/OS , vous pouvez en faire une file d'attente partagée ou privée.

Une file d'attente est considérée par un programme comme étant *locale* si elle est détenue par le gestionnaire de files d'attente auquel le programme est connecté. Vous pouvez extraire et insérer des messages dans les files d'attente locales.

L'objet définition de file d'attente stocke les informations de définition de la file d'attente, ainsi que les messages physiques insérés dans la file d'attente.

### **file d'attente du gestionnaire de files d'attente local**

La file d'attente existe sur le gestionnaire de files d'attente local.

 La file d'attente est appelée file d'attente privée sur z/OS.

### **File d'attente partagée (z/OS uniquement)**

La file d'attente existe dans un référentiel partagé accessible à tous les gestionnaires de files d'attente appartenant au groupe de partage de files d'attente propriétaire du référentiel partagé.

Les applications connectées à un gestionnaire de files d'attente dans le groupe de partage de files d'attente peuvent placer des messages dans des files d'attente de ce type et en supprimer. Ces files d'attente sont en fait les mêmes que les files d'attente locales. La valeur de l'attribut de file d'attente **QType** est MQQT\_LOCAL.

Les applications connectées au gestionnaire de files d'attente local peuvent placer des messages dans des files d'attente de ce type et en supprimer. La valeur de l'attribut de file d'attente **QType** est MQQT\_LOCAL.

### File d'attente de cluster

Vous pouvez stocker des messages dans une file d'attente de cluster sur le gestionnaire de files d'attente où ils sont définis. Une file d'attente de cluster est une file d'attente hébergée par un gestionnaire de files d'attente de cluster et accessible aux autres gestionnaires de files d'attente dans le cluster. La valeur de l'attribut de file d'attente **QType** est MQQT\_CLUSTER.

Une file d'attente de cluster est annoncée aux autres gestionnaires de files d'attente de cluster. Ces autres gestionnaires de files d'attente de cluster peuvent insérer des messages dans une file d'attente de cluster sans qu'une définition de file d'attente éloignée correspondante soit nécessaire. Une file d'attente de cluster peut être annoncée dans plusieurs clusters à l'aide d'une liste de noms de cluster.

Lorsqu'une file d'attente est annoncée, les gestionnaires de files d'attente de cluster peuvent y insérer des messages. Pour insérer un message, le gestionnaire de files d'attente doit déterminer, à partir des référentiels complets, l'emplacement où la file d'attente est hébergée. Il ajoute alors des informations de routage au message et insère ce dernier dans une file d'attente de transmission du cluster.

Un gestionnaire de files d'attente peut stocker des messages pour d'autres gestionnaires d'un cluster dans plusieurs files d'attente de transmission. Vous pouvez configurer un gestionnaire de files d'attente de deux manières pour stocker les messages dans plusieurs files d'attente de transmission de cluster. Si vous définissez l'attribut de gestionnaire de files d'attente **DEFCLXQ** sur CHANNEL, une file d'attente de transmission de cluster différente est créée automatiquement à partir de SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE pour chaque canal émetteur de cluster. Si vous définissez l'option de file d'attente de transmission CLCHNAME sur un ou plusieurs canaux émetteurs de cluster, le gestionnaire de files d'attente peut stocker les messages pour les canaux correspondants dans cette file d'attente de transmission.



**Avertissement :** Si vous utilisez SYSTEM.CLUSTER.TRANSMIT.QUEUES avec un gestionnaire de files d'attente mis à niveau à partir d'une version du produit antérieure à IBM WebSphere MQ 7.5, vérifiez que SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE a l'option SHARE/NOSHARE définie sur **SHARE**.



Une file d'attente de cluster peut être une file d'attente partagée par les membres d'un groupe de partage de files d'attente dans IBM MQ for z/OS.

### File d'attente éloignée

Une file d'attente éloignée n'est pas une file d'attente physique ; il s'agit de la définition locale d'une file d'attente qui existe sur un gestionnaire de files d'attente éloignées. La définition locale de la file d'attente éloignée contient des informations indiquant au gestionnaire de files d'attente local comment acheminer les messages vers le gestionnaire de files d'attente éloignées.

Les applications connectées au gestionnaire de files d'attente local peuvent placer des messages dans des files d'attente de ce type ; les messages sont placés dans la file d'attente de transmission locale utilisée pour acheminer les messages vers le gestionnaire de files d'attente éloignées. Les applications ne peuvent pas supprimer les messages des files d'attente éloignées. La valeur de l'attribut de file d'attente **QType** est MQQT\_REMOTE.

Vous pouvez également utiliser une définition de file d'attente éloignée pour:

- Alias de file d'attente de réponses

Dans ce cas, le nom de la définition est le nom d'une file d'attente de réponse. Pour plus d'informations, voir [Alias de file d'attente de réponse et clusters](#).

- Alias du gestionnaire de files d'attente

Dans ce cas, le nom de la définition est un alias pour un gestionnaire de files d'attente et non le nom d'une file d'attente. Pour plus d'informations, voir [Alias des gestionnaires de files d'attente et clusters](#).

### File d'attente d'alias

Il ne s'agit pas d'une file d'attente physique ; il s'agit d'un autre nom pour une file d'attente locale, une file d'attente partagée, une file d'attente de cluster ou une file d'attente éloignée. Le nom de la file d'attente dans laquelle l'alias est résolu fait partie de la définition de la file d'attente alias.

Les applications connectées au gestionnaire de files d'attente local peuvent placer des messages dans des files d'attente de ce type ; les messages sont placés dans la file d'attente dans laquelle l'alias est résolu. Les applications peuvent supprimer des messages des files d'attente de ce type si l'alias est résolu en file d'attente locale, en file d'attente partagée ou en file d'attente de cluster comportant une instance locale. La valeur de l'attribut de file d'attente **QType** est MQQT\_ALIAS.

### File d'attente modèle

Il ne s'agit pas d'une file d'attente physique ; il s'agit d'un ensemble d'attributs de file d'attente à partir desquels une file d'attente locale peut être créée.

Les messages ne peuvent pas être stockés dans des files d'attente de ce type.

## limites de file d'attente

Depuis la IBM MQ 9.2.0, vous avez la possibilité de configurer et de surveiller des files d'attente qui prendront en charge une quantité nettement supérieure à la limite par défaut de deux téraoctets utilisée dans les versions antérieures d' IBM MQ. Vous pouvez également diminuer la taille maximale d'un fichier de file d'attente.

Pour vous permettre de configurer des files d'attente, vous pouvez utiliser l'attribut **MAXFSIZE** sur les files d'attente locales et modèles, et pour surveiller les files d'attente, vous pouvez utiliser les attributs de statut de file d'attente **CURFSIZE** et **CURMAXFS**.

Pour plus d'informations, voir [Modifying IBM MQ queue files](#).

## Attributs File d'attente

Certains attributs de file d'attente s'appliquent à tous les types de file d'attente ; d'autres s'appliquent uniquement à certains types de file d'attente. Les types de file d'attente auxquels un attribut s'applique sont affichés dans [Tableau 561](#), à la [page 878](#) et dans les tableaux suivants.

Le [Tableau 561](#), à la [page 878](#) récapitule les attributs spécifiques aux files d'attente. Les attributs sont décrits par ordre alphabétique.

**Remarque :** Les noms des attributs affichés dans cette section sont des noms descriptifs utilisés avec les appels MQINQ et MQSET ; les noms sont identiques à ceux des commandes PCF. Lorsque des commandes MQSC sont utilisées pour définir, modifier ou afficher des attributs, des noms abrégés alternatifs sont utilisés ; voir [Commandes MQSC](#) pour plus de détails.

Dans le tableau suivant, les colonnes s'appliquent comme suit:

- La colonne des files d'attente locales s'applique également aux files d'attente partagées.
- La colonne des files d'attente modèles indique les attributs hérités par la file d'attente locale créée à partir de la file d'attente modèle.
- La colonne des files d'attente de cluster indique les attributs qui peuvent être renseignés lorsque la file d'attente de cluster est ouverte pour l'interrogation seule ou pour l'interrogation et la sortie. Si d'autres attributs sont requis, l'appel renvoie le code achèvement MQCC\_WARNING et le code anomalie MQRC\_SELECTOR\_NOT\_FOR\_TYPE (2068).

Si la file d'attente de cluster est ouverte pour l'interrogation plus une ou plusieurs entrées, parcourez ou définissez, la colonne des files d'attente locales s'applique à la place.

Si la file d'attente de cluster est ouverte pour l'interrogation seule ou pour l'interrogation et la sortie, en plus de la spécification du nom du gestionnaire de files d'attente de base, la colonne des files d'attente locales s'applique à la place.

Tableau 561. Attributs des files d'attente

Attribut	Description	Locale	Modèle	Alias	Eloigné	Cluster
<u>AlterationDate</u>	Date de la dernière modification de la définition	X		X	X	
<u>AlterationTime</u>	Heure de la dernière modification de la définition	X		X	X	
<u>BackoutRequeueQName</u>	Nom de file d'attente de remise en file d'attente d'annulation excessive	X	X			
<u>BackoutThreshold</u>	Seuil d'annulation	X	X			
<u>BaseQName</u>	Nom de la file d'attente dans laquelle l'alias est résolu			X		
<u>CFStrucName</u>	Nom de la structure d'unité de couplage	X	X			
<u>CLCHNAME</u>	Noms des canaux émetteurs de cluster	✓	✓			
<u>ClusterName</u>	Nom du cluster auquel appartient la file d'attente	X		X	X	X
<u>ClusterNameList</u>	Nom de l'objet liste de noms contenant les noms des clusters auxquels appartient la file d'attente	X		X	X	
<u>CLWLQueuePriority</u>	Priorité de file d'attente de charge de travail	X		X	X	X
<u>CLWLQueueRank</u>	Rang de file d'attente de charge de travail	X		X	X	X
<u>CLWLUseQ</u>	Utiliser la file d'attente éloignée	X				
<u>CreationDate</u>	Date de création de la file d'attente	X				
<u>CreationTime</u>	Heure de création de la file d'attente	X				
<u>CurrentQDepth</u>	Longueur actuelle de la file	X				
<u>DefaultPutResponse</u>	Réponse d'insertion par défaut	✓	✓	✓	✓	
<u>DefBind</u>	Liaison par défaut	X		X	X	X
<u>DefinitionType attribute</u>	Type de définition de file d'attente	X	X			
<u>DefInputOpenOption</u>	Option d'ouverture en entrée par défaut	X	X			
<u>DefPersistence</u>	Persistance par défaut	X	X	X	X	X
<u>DefPriority</u>	Priorité par défaut	✓	✓	✓	✓	✓
<u>DefReadAhead</u>	Lecture anticipée par défaut	X	X	X		
<u>DistLists</u>	Prise en charge de la liste de diffusion	X	X			
<u>HardenGetBackout</u>	Indique s'il convient de conserver un nombre d'annulations précis	X	X			
<u>IndexType</u>	Type d'index	X	X			
<u>InhibitGet</u>	Indique si les opérations d'extraction de la file d'attente sont autorisées	X	X	X		

Tableau 561. Attributs des files d'attente (suite)

Attribut	Description	Locale	Modèle	Alias	Eloigné	Cluster
<a href="#">InhibitPut</a>	Indique si les opérations d'insertion pour la file d'attente sont autorisées	X	X	X	X	X
<a href="#">InitiationQName</a>	Nom de la file d'attente d'initialisation	X	X			
<a href="#">MaxMsgLength</a>	Longueur maximale des messages (en octets)	X	X			
<a href="#">MaxQDepth</a>	Long. max. file d'attente	X	X			
<a href="#">MsgDeliverySequence attribute</a>	Séquence livraison messages	X	X			
<a href="#">NonPersistentMessage Class</a>	Objectif de fiabilité pour les messages non persistants	X	X			
<a href="#">OpenInputCount</a>	Nombre d'ouvertures pour la saisie	X				
<a href="#">OpenOutputCount</a>	Nombre d'ouvertures pour la sortie	X				
<a href="#">PropertyControl</a>	Contrôle des propriétés	✓	✓	✓		
<a href="#">ProcessName</a>	Nom du processus	X	X			
<a href="#">QDepthHighEvent attribute</a>	Indique si les événements Longueur élevée de la file d'attente sont générés	X	X			
<a href="#">QDepthHighLimit</a>	Limite supérieure de la longueur de la file d'attente	X	X			
<a href="#">QDepthLowEvent attribute</a>	Indique si les événements Longueur faible de la file d'attente sont générés	X	X			
<a href="#">QDepthLowLimit attribute</a>	Limite inférieure de la longueur de la file d'attente	X	X			
<a href="#">QDepthMaxEvent</a>	Indique si des événements de file d'attente saturée sont générés	X	X			
<a href="#">QDesc</a>	Description de file d'attente	X	X	X	X	X
<a href="#">QName</a>	Nom de la file d'attente	X		X	X	X
<a href="#">QServiceInterval</a>	Cible de l'intervalle de service de file d'attente	X	X			
<a href="#">QServiceIntervalEvent attribute</a>	Indique si des événements d'intervalle de service élevé ou d'intervalle de service OK sont générés	X	X			
<a href="#">QSGDisp attribute</a>	Disposition d'un groupe de partage de files d'attente	X		X	X	
<a href="#">QueueAccounting</a>	Collecte des données de comptabilité des files d'attente	X	X	X	X	X
<a href="#">QueueMonitoring</a>	Données de surveillance en ligne pour les files d'attente	X	✓			
<a href="#">QueueStatistics</a>	collecte de données statistiques de file d'attente	X	X	X	X	X
<a href="#">QType</a>	Type de file d'attente	X		X	X	X
<a href="#">RemoteQMgrName</a>	Nom du gestionnaire de files d'attente éloignées				X	
<a href="#">RemoteQName</a>	Nom de la file d'attente éloignée				X	
<a href="#">RetentionInterval</a>	Intervalle de conservation	X	X			

Attribut	Description	Locale	Modèle	Alias	Eloigné	Cluster
<u>Scope</u>	Indique si une entrée de la file d'attente existe également dans un répertoire de cellule	X		X	X	
<u>Shareability</u>	Partageabilité de la file d'attente	X	X			
<u>StorageClass</u>	Classe de stockage pour la file d'attente	X	X			
<u>TriggerControl</u>	Contrôle du déclenchement	X	X			
<u>TriggerData</u>	Données de déclenchement	X	X			
<u>TriggerDepth</u>	Longueur de déclenchement	X	X			
<u>TriggerMsgPriority</u>	Priorité des messages de seuil des déclencheurs	X	X			
<u>TriggerType</u>	Type de déclenchement	X	X			
<u>Usage attribute</u>	Utilisation de la file d'attente	X	X			
<u>XmitQName</u>	Nom de la file d'attente de transmission				X	

### Concepts associés

[Files d'attente de cluster](#)

[Files d'attente locales](#)

[Comment choisir le type de file d'attente de transmission de cluster à utiliser](#)

### **AlterationDate (MQCHAR12)**

Date de la dernière modification de la définition.

Locale	Modèle	Alias	Eloigné	Cluster
X		X	X	

Il s'agit de la date de la dernière modification de la définition. Le format de la date est YYYY-MM-DD, complété par deux blancs de fin pour obtenir une longueur de 12 octets (par exemple, 1992-09-23↵↵, où ↵↵ représente deux caractères blancs).

Les valeurs de certains attributs (par exemple, *CurrentQDepth*) changent lorsque le gestionnaire de files d'attente fonctionne. Les modifications apportées à ces attributs n'affectent pas *AlterationDate*.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_ALTERATION\_DATE avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_DATE\_LENGTH.

### **AlterationTime (MQCHAR8)**

Heure de la dernière modification de la définition.

Locale	Modèle	Alias	Eloigné	Cluster
X		X	X	



Il s'agit de l'heure à laquelle la définition a été modifiée pour la dernière fois. Le format de l'heure est HH.MM.SS à l'aide de l'horloge au format 24 heures, avec un zéro non significatif si l'heure est inférieure à 10 (par exemple, 09.10.20).

- Sous z/OS, l'heure est GMT (Greenwich Mean Time), sous réserve que l'horloge système soit réglée avec précision sur GMT.
- Dans d'autres environnements, l'heure est l'heure locale.

Les valeurs de certains attributs (par exemple, *CurrentQDepth*) changent lorsque le gestionnaire de files d'attente fonctionne. Les modifications apportées à ces attributs n'affectent pas *AlterationTime*.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_ALTERATION\_TIME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_TIME\_LENGTH.

### **BackoutRequeueQName (MQCHAR48)**

Il s'agit du nom de la file d'attente de remise en file d'attente d'annulation excessive. En plus de permettre l'interrogation de sa valeur, le gestionnaire de files d'attente n'effectue aucune action en fonction de la valeur de cet attribut.

Tableau 564. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Les applications qui s'exécutent dans WebSphere Application Server et celles qui utilisent IBM MQ Application Server Facilities utilisent cet attribut pour déterminer l'emplacement des messages qui ont été annulés. Pour toutes les autres applications, le gestionnaire de files d'attente n'effectue aucune action en fonction de la valeur de l'attribut.

IBM MQ classes for JMS utilise cet attribut pour déterminer où transférer un message qui a déjà été annulé le nombre maximal de fois spécifié par l'attribut *BackoutThreshold*.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_BACKOUT\_REQ\_Q\_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_Q\_NAME\_LENGTH.

### **BackoutThreshold (MQLONG)**

Il s'agit du seuil d'annulation. En plus de permettre l'interrogation de sa valeur, le gestionnaire de files d'attente n'effectue aucune action en fonction de la valeur de cet attribut.

Tableau 565. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Les applications exécutées dans WebSphere Application Server et celles qui utilisent IBM MQ Application Server Facilities utiliseront cet attribut pour déterminer si un message doit être annulé. Pour toutes les autres applications, le gestionnaire de files d'attente n'effectue aucune action en fonction de la valeur de l'attribut.

IBM MQ classes for JMS utilise cet attribut pour déterminer le nombre de fois où un message peut être annulé avant d'être transféré dans la file d'attente spécifiée par l'attribut *BackoutRequeueQName*.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_BACKOUT\_THRESHOLD avec l'appel MQINQ.

### **BaseQName (MQCHAR48)**

Il s'agit du nom d'une file d'attente définie pour le gestionnaire de files d'attente local.

Tableau 566. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
		X		

(Pour plus d'informations sur les noms de file d'attente, voir la zone [MQOD- ObjectName](#).) La file d'attente est de l'un des types suivants:

#### **MQQT\_LOCAL**

File d'attente locale.

#### **MQQT\_REMOTE**

Définition locale d'une file d'attente éloignée.

#### **MQQT\_CLUSTER**

File d'attente de cluster.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_BASE\_Q\_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_Q\_NAME\_LENGTH.

#### **BaseType (MQCFIN)**

Type d'objet auquel l'alias est résolu.

Tableau 567. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
		X		

Ses valeurs sont l'une des suivantes :

#### **MQOT\_Q**

Le type d'objet de base est une file d'attente

#### **MQOT\_TOPIC**

Le type d'objet de base est une rubrique

#### **CFStrucName (MQCHAR12)**


Il s'agit du nom de la structure d'unité de couplage dans laquelle les messages de la file d'attente sont stockés. Le premier caractère du nom est compris entre A et Z, et les caractères restants sont compris entre A et Z, 0 et 9, ou à blanc.

Tableau 568. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Pour obtenir le nom complet de la structure dans l'unité de couplage, ajoutez la valeur de l'attribut de gestionnaire de files d'attente **QSGName** à la valeur de l'attribut de file d'attente **CFStrucName** .

Cet attribut s'applique uniquement aux files d'attente partagées ; il est ignoré si *QSGD*.sp ne possède pas la valeur MQQSGD\_SHARED.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_CF\_STRUC\_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_CF\_STRUC\_NAME\_LENGTH.

 Cet attribut est pris en charge uniquement sous z/OS.

### **ClusterChannelNom (MQCHAR20)**

ClusterChannelNom est le nom générique des canaux émetteurs de cluster qui utilisent cette file d'attente comme file d'attente de transmission. L'attribut indique quels canaux émetteurs de cluster envoient des messages à un canal récepteur de cluster à partir de cette file d'attente de transmission de cluster.

Tableau 569. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

La configuration du gestionnaire de files d'attente par défaut s'applique à tous les canaux émetteurs de cluster pour envoyer des messages à partir d'une seule file d'attente de transmission, SYSTEM.CLUSTER.TRANSMIT.QUEUE. La configuration par défaut peut être modifiée en modifiant l'attribut du gestionnaire de files d'attente, **DefClusterXmitQueueType**. La valeur par défaut de l'attribut est SCTQ. Vous pouvez la remplacer par CHANNEL. Si vous définissez l'attribut **DefClusterXmitQueueType** sur CANAL, chaque canal émetteur de cluster utilise par défaut une file d'attente de transmission de cluster spécifique, SYSTEM.CLUSTER.TRANSMIT.ChannelName.

Vous pouvez aussi associer l'attribut de file d'attente de transmission ClusterChannelName à un canal émetteur de cluster manuellement. Les messages destinés au gestionnaire de files d'attente connecté par le canal émetteur de cluster sont stockés dans la file d'attente de transmission qui identifie le canal émetteur de cluster. Ils ne sont pas stockés dans la file d'attente de transmission du cluster. Si vous associez l'attribut ClusterChannelName à des blancs, le canal utilise la file d'attente de transmission du cluster par défaut lorsqu'il redémarre. La file d'attente par défaut est SYSTEM.CLUSTER.TRANSMIT.ChannelName ou SYSTEM.CLUSTER.TRANSMIT.QUEUE, en fonction de la valeur de l'attribut DefClusterXmitQueueType du gestionnaire de files d'attente.

En spécifiant des astérisques, "\*", dans **ClusterChannelName**, vous pouvez associer une file d'attente de transmission à un ensemble de canaux émetteurs de cluster. Ces astérisques peuvent se trouver au début, à la fin ou à plusieurs endroits dans la chaîne de nom de canal. **ClusterChannelName** est limité à une longueur de 20 caractères : MQ\_CHANNEL\_NAME\_LENGTH.

### **ClusterName (MQCHAR48)**

Il s'agit du nom du cluster auquel appartient la file d'attente.

Tableau 570. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X		X	X	X

Si la file d'attente appartient à plusieurs clusters, *ClusterNameList* spécifie le nom d'un objet liste de noms qui identifie les clusters et *ClusterName* est vide. Au moins l'un des éléments *ClusterName* et *ClusterNameList* doit être vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_CLUSTER\_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_CLUSTER\_NAME\_LENGTH.

### **ClusterNameList (MQCHAR48)**

Il s'agit du nom d'un objet liste de noms qui contient les noms des clusters auxquels appartient cette file d'attente.

Tableau 571. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X		X	X	

Si la file d'attente appartient à un seul cluster, l'objet liste de noms ne contient qu'un seul nom. Vous pouvez également utiliser *ClusterName* pour spécifier le nom du cluster, auquel cas *ClusterNameList* est vide. Au moins l'un des éléments *ClusterName* et *ClusterNameList* doit être vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_CLUSTER\_NAMELIST avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_NAMELIST\_NAME\_LENGTH.

### **CLWLQueuePriority (MQLONG)**

Il s'agit de la priorité de la file d'attente de charge de travail du cluster, une valeur comprise entre 0 et 9 représentant la priorité de la file d'attente.

<i>Tableau 572. Types de file d'attente auxquels cet attribut s'applique</i>				
Locale	Modèle	Alias	Eloigné	Cluster
X		X	X	X

Pour plus d'informations, voir [Files d'attente de cluster](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_CLWL\_Q\_PRIORITY avec l'appel MQINQ.

### **CLWLQueueRank (MQLONG)**

Il s'agit du rang de la file d'attente de charge de travail du cluster, une valeur comprise entre 0 et 9 représentant le rang de la file d'attente.

<i>Tableau 573. Types de file d'attente auxquels cet attribut s'applique</i>				
Locale	Modèle	Alias	Eloigné	Cluster
X		X	X	X

Pour plus d'informations, voir [Files d'attente de cluster](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_CLWL\_Q\_RANK avec l'appel MQINQ.

### **CLWLUseQ (MQLONG)**

Cela définit le comportement d'un MQPUT lorsque la file d'attente cible comporte à la fois une instance locale et au moins une instance de cluster distant. Si l'insertion émane d'un canal de cluster, cet attribut ne s'applique pas.

<i>Tableau 574. Types de file d'attente auxquels cet attribut s'applique</i>				
Locale	Modèle	Alias	Eloigné	Cluster
X				

La valeur est l'une des suivantes :

#### **MQCLWL\_USEQ\_ANY**

Utilisez des files d'attente distantes et locales.

#### **MQCLWL\_USEQ\_LOCAL**

N'utilisez pas de files d'attente éloignées.

#### **MQCLWL\_USEQ\_AS\_Q\_MGR**

Hérite de la définition de MQIA\_CLWL\_USEQ du gestionnaire de files d'attente.

Pour plus d'informations, voir [Files d'attente de cluster](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_CLWL\_USEQ avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_CLWL\_USEQ\_LENGTH.

### **CreationDate (MQCHAR12)**

Il s'agit de la date de création de la file d'attente.

Tableau 575. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X				

Le format de la date est YYYY-MM-DD, complété par deux blancs de fin pour obtenir une longueur de 12 octets (par exemple, 2013-09-23 , où représente 2 caractères blancs).

- Sous IBM i, la date de création d'une file d'attente peut être différente de celle de l'entité de système d'exploitation sous-jacente (fichier ou espace utilisateur) qui représente la file d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_CREATION\_DATE avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_CREATION\_DATE\_LENGTH.

### **CreationTime (MQCHAR8)**

Il s'agit de l'heure à laquelle la file d'attente a été créée.

Tableau 576. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X				

Le format de l'heure est HH.MM.SS à l'aide de l'horloge au format 24 heures, avec un zéro non significatif si l'heure est inférieure à 10 (par exemple, 09.10.20).

- Sous z/OS, l'heure est GMT (Greenwich Mean Time), sous réserve que l'horloge système soit réglée avec précision sur GMT.
- Dans d'autres environnements, l'heure est l'heure locale.
- Sous IBM i, l'heure de création d'une file d'attente peut être différente de celle de l'entité de système d'exploitation sous-jacente (fichier ou espace utilisateur) qui représente la file d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_CREATION\_TIME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_CREATION\_TIME\_LENGTH.

### **CurrentQDepth (MQLONG)**

Il s'agit du nombre de messages stockés actuellement dans la file d'attente.

Tableau 577. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X				

Elle est incrémentée lors d'un appel MQPUT et lors de l'annulation d'un appel MQGET. Il est décrémenté lors d'un appel MQGET non navigant et lors de l'annulation d'un appel MQPUT. Cela a pour effet que le nombre inclut les messages qui ont été placés dans la file d'attente dans une unité d'oeuvre, mais qui n'ont pas encore été validés, même s'ils ne peuvent pas être extraits par l'appel MQGET. De même, il exclut les messages qui ont été extraits dans une unité d'oeuvre à l'aide de l'appel MQGET, mais qui n'ont pas encore été validés.

Ce nombre inclut également les messages dont l'heure d'expiration a été dépassée mais qui n'ont pas encore été supprimés, bien que ces messages ne puissent pas être extraits. Pour plus d'informations, voir [MQMD-Zone d'expiration](#).

Le traitement de l'unité de travail et la segmentation des messages peuvent tous deux entraîner un dépassement de la valeur de *CurrentQDepth* *MaxQDepth*. Toutefois, cela n'affecte pas la possibilité d'extraction des messages ; tous les messages de la file d'attente peuvent être extraits à l'aide de l'appel MQGET de manière normale.

La valeur de cet attribut fluctue au fur et à mesure que le gestionnaire de files d'attente fonctionne. Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_CURRENT\_Q\_DEPTH avec l'appel MQINQ.

### Réponse DefaultPut(MQLONG)

Indique le type de réponse à utiliser pour les opérations d'insertion dans la file d'attente lorsqu'une application spécifie MQPMO\_RESPONSE\_AS\_Q\_DEF.

Tableau 578. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X	X	X	

Ses valeurs sont l'une des suivantes :

#### MQPRT\_SYNC\_REPONSE

L'opération d'insertion est émise de manière synchrone et renvoie une réponse.

#### MQPRT\_ASYNC\_RESPONSE

L'opération d'insertion est émise de manière asynchrone et renvoie un sous-ensemble de zones MQMD.

### DefBind (MQLONG)

Il s'agit de la liaison par défaut utilisée lorsque MQOO\_BIND\_AS\_Q\_DEF est spécifié dans l'appel MQOPEN et que la file d'attente est une file d'attente de cluster.

Tableau 579. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X		X	X	X

La valeur est l'une des suivantes :

#### MQBND\_BIND\_ON\_OPEN

Liaison corrigée par l'appel MQOPEN.

#### MQBND\_BIND\_NOT\_FIXED

Liaison non fixe.

#### MQBND\_BIND\_ON\_GROUPE

Permet à une application de demander qu'un groupe de messages soit alloué à la même instance de destination.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_DEF\_BIND avec l'appel MQINQ.

### DefinitionType (MQLONG)

Indique comment la file d'attente a été définie.

Tableau 580. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

La valeur est l'une des suivantes :

#### MQQDT\_PRÉDÉFINI

La file d'attente est une file d'attente permanente créée par l'administrateur système ; seul l'administrateur système peut la supprimer.

Les files d'attente prédéfinies sont créées à l'aide de la commande DEFINE MQSC et peuvent être supprimées uniquement à l'aide de la commande DELETE MQSC. Les files d'attente prédéfinies ne peuvent pas être créées à partir de files d'attente modèles.

Les commandes peuvent être émises par un opérateur ou par un utilisateur autorisé à envoyer un message de commande à la file d'attente d'entrée de commande (pour plus d'informations, voir Attribut QNameCommandInput ).

#### **MQQDT\_PERMANENT\_DYNAMIQUE**

La file d'attente est une file d'attente permanente créée par une application émettant un appel MQOPEN avec le nom d'une file d'attente modèle indiquée dans le descripteur d'objet MQOD. La définition de file d'attente modèle avait la valeur MQQDT\_PERMANENT\_DYNAMIC pour l'attribut **DefinitionType** .

Ce type de file d'attente peut être supprimé à l'aide de l'appel MQCLOSE. Pour plus d'informations, voir «MQCLOSE-Fermer l'objet», à la page 679.

La valeur de l'attribut **QSGDisp** pour une file d'attente dynamique permanente est MQQSGD\_Q\_MGR.

#### **MQQDT\_TEMPORARY\_DYNAMIQUE**

La file d'attente est une file d'attente temporaire créée par une application qui émet un appel MQOPEN avec le nom d'une file d'attente modèle indiquée dans le descripteur d'objet MQOD. La définition de file d'attente modèle avait la valeur MQQDT\_TEMPORARY\_DYNAMIC pour l'attribut **DefinitionType** .

Ce type de file d'attente est supprimé automatiquement par l'appel MQCLOSE lorsqu'il est fermé par l'application qui l'a créé.

La valeur de l'attribut **QSGDisp** pour une file d'attente dynamique temporaire est MQQSGD\_Q\_MGR.

#### **MQQDT\_PARTAGE\_DYNAMIQUE**

La file d'attente est une file d'attente permanente partagée qui a été créée par une application émettant un appel MQOPEN avec le nom d'une file d'attente modèle indiquée dans le descripteur d'objet MQOD. La définition de file d'attente modèle avait la valeur MQQDT\_SHARED\_DYNAMIC pour l'attribut **DefinitionType** .

Ce type de file d'attente peut être supprimé à l'aide de l'appel MQCLOSE. Pour plus d'informations, voir «MQCLOSE-Fermer l'objet», à la page 679.

La valeur de l'attribut **QSGDisp** pour une file d'attente dynamique partagée est MQQSGD\_SHARED.

Cet attribut dans une définition de file d'attente modèle n'indique pas comment la file d'attente modèle a été définie, car les files d'attente modèles sont toujours prédéfinies. A la place, la valeur de cet attribut dans la file d'attente modèle est utilisée pour déterminer le *DefinitionType* de chacune des files d'attente dynamiques créées à partir de la définition de file d'attente modèle à l'aide de l'appel MQOPEN.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_DEFINITION\_TYPE avec l'appel MQINQ.

#### **DefInputOpenOption (MQLONG)**

Il s'agit de la méthode par défaut permettant d'ouvrir la file d'attente pour l'entrée.

<i>Tableau 581. Types de file d'attente auxquels cet attribut s'applique</i>				
<b>Locale</b>	<b>Modèle</b>	<b>Alias</b>	<b>Eloigné</b>	<b>Cluster</b>
X	X			

Elle s'applique si l'option MQOO\_INPUT\_AS\_Q\_DEF est spécifiée dans l'appel MQOPEN lorsque la file d'attente est ouverte. La valeur est l'une des suivantes :

#### **MQOO\_INPUT\_EXCLUSIVE**

Ouvrez la file d'attente pour obtenir les messages avec un accès exclusif.

La file d'attente est ouverte pour être utilisée avec les appels MQGET suivants. L'appel échoue avec le code anomalie MQRC\_OBJECT\_IN\_USE si la file d'attente est actuellement ouverte par cette application ou par une autre application pour l'entrée d'un type quelconque (MQOO\_INPUT\_SHARED ou MQOO\_INPUT\_EXCLUSIVE).

## **MQOO\_INPUT\_SHARED**

Ouvrez la file d'attente pour obtenir les messages avec accès partagé.

La file d'attente est ouverte pour être utilisée avec les appels MQGET suivants. L'appel peut aboutir si la file d'attente est actuellement ouverte par cette application ou par une autre application avec MQOO\_INPUT\_SHARED, mais échoue avec le code anomalie MQRC\_OBJECT\_IN\_USE si la file d'attente est actuellement ouverte avec MQOO\_INPUT\_EXCLUSIVE.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_DEF\_INPUT\_OPEN\_OPTION avec l'appel MQINQ.

## **DefPersistence (MQLONG)**

Il s'agit de la persistance par défaut des messages dans la file d'attente. Elle s'applique si MQPER\_PERSISTENCE\_AS\_Q\_DEF est spécifié dans le descripteur de message lors de l'insertion du message.

<i>Tableau 582. Types de file d'attente auxquels cet attribut s'applique</i>				
<b>Locale</b>	<b>Modèle</b>	<b>Alias</b>	<b>Eloigné</b>	<b>Cluster</b>
X	X	X	X	X

S'il existe plusieurs définitions dans le chemin de résolution de nom de file d'attente, la persistance par défaut est extraite de la valeur de cet attribut dans la *première* définition du chemin au moment de l'appel MQPUT ou MQPUT1 . Possibilités :

- Une file d'attente alias
- Une file d'attente locale
- Définition locale d'une file d'attente éloignée
- Un alias de gestionnaire de files d'attente
- Une file d'attente de transmission (par exemple, la file d'attente *DefXmitQName* )

La valeur est l'une des suivantes :

## **MQPER\_PERSISTANT**

Le message survit aux échecs du système et aux redémarrages du gestionnaire de files d'attente. Les messages persistants ne peuvent pas être placés sur:

- Files d'attente dynamiques temporaires
- Files d'attente partagées qui sont mappées à un objet CFSTRUCT au niveau CFLEVEL (2) ou à un niveau inférieur, ou où l'objet CFSTRUCT est défini en tant que RECOVER (NO).

Les messages persistants peuvent être placés dans des files d'attente dynamiques permanentes et dans des files d'attente prédéfinies.

## **MQPER\_NON\_PERSISTENT**

Le message ne survit normalement pas aux défaillances du système ou aux redémarrages du gestionnaire de files d'attente. Cela s'applique même si une copie intacte du message est trouvée sur la mémoire secondaire lors du redémarrage d'un gestionnaire de files d'attente.

Dans le cas de files d'attente partagées, les messages non persistants *survivent* aux redémarrages des gestionnaires de files d'attente dans le groupe de partage de files d'attente, mais ne survivent pas aux échecs de l'unité de couplage utilisée pour stocker les messages dans les files d'attente partagées.

Les messages persistants et non persistants peuvent exister dans la même file d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_DEF\_PERSISTENCE avec l'appel MQINQ.



### **DefPriority (MQLONG)**

Il s'agit de la priorité par défaut pour les messages de la file d'attente. Cela s'applique si MQPRI\_PRIORITY\_AS\_Q\_DEF est spécifié dans le descripteur de message lorsque le message est inséré dans la file d'attente.

Tableau 583. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X	X	X	X

S'il existe plusieurs définitions dans le chemin de résolution de nom de file d'attente, la priorité par défaut du message est extraite de la valeur de cet attribut dans la *première* définition du chemin au moment de l'opération d'insertion. Possibilités :

- Une file d'attente alias
- Une file d'attente locale
- Définition locale d'une file d'attente éloignée
- Un alias de gestionnaire de files d'attente
- Une file d'attente de transmission (par exemple, la file d'attente *DefXmitQName* )

La manière dont un message est placé dans une file d'attente dépend de la valeur de l'attribut **MsgDeliverySequence** de la file d'attente:

- Si l'attribut **MsgDeliverySequence** est MQMDS\_PRIORITY, la position logique à laquelle un message est placé dans la file d'attente dépend de la valeur de la zone *Priority* dans le descripteur de message.
- Si l'attribut **MsgDeliverySequence** est MQMDS\_FIFO, les messages sont placés dans la file d'attente comme s'ils avaient une priorité égale à *DefPriority* de la file d'attente résolue, quelle que soit la valeur de la zone *Priority* dans le descripteur de message. Toutefois, la zone *Priority* conserve la valeur spécifiée par l'application qui a inséré le message. Pour plus d'informations, voir [Attribut de séquenceMsgDelivery](#) .

Les priorités sont comprises entre zéro (valeur la plus faible) et *MaxPriority* (valeur la plus élevée) ; voir [AttributMaxPriority](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_DEF\_PRIORITY avec l'appel MQINQ.

### **DefReadAhead (MQLONG)**

Indique le comportement de lecture anticipée par défaut pour les messages non persistants distribués au client.

Tableau 584. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X	X		

DefReadAhead peut être défini sur l'une des valeurs suivantes:

#### **MQREADA\_NO**

Les messages non persistants ne sont pas envoyés au client avant qu'une application ne les demande. Un maximum d'un message non persistant peut être perdu si le client se termine de manière anormale.

#### **MQREADA\_YES**

Les messages non persistants sont envoyés au client avant qu'une application ne les demande. Les messages non persistants peuvent être perdus si le client se termine de manière anormale ou si le client ne consomme pas tous les messages qu'il envoie.

## **MQREADA\_DISABLED**

La lecture anticipée des messages non persistants n'est pas activée pour cette file d'attente. Les messages ne sont pas envoyés à l'avance au client, que la lecture anticipée soit demandée ou non par l'application client.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_DEF\_READ\_AHEAD avec l'appel MQINQ.

## **DefPResp (MQLONG)**

L'attribut de type de réponse d'insertion par défaut (DEFPRESP) définit la valeur utilisée par les applications lorsque le type PutResponse dans MQPMO a été défini sur MQPMO\_RESPONSE\_AS\_Q\_DEF. Cet attribut est valide pour tous les types de file d'attente.

Tableau 585. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X	X	X	X

La valeur est l'une des suivantes :

### **SYNC**

L'opération d'insertion est émise de manière synchrone et renvoie une réponse.

### **ASYN**

L'opération d'insertion est émise de manière asynchrone et renvoie un sous-ensemble de zones MQMD.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_DEF\_PUT\_RESPONSE\_TYPE avec l'appel MQINQ.

## **DistLists (MQLONG)**

Indique si les messages de liste de distribution peuvent être placés dans la file d'attente.

Tableau 586. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Un agent MCA définit l'attribut pour indiquer au gestionnaire de files d'attente local si le gestionnaire de files d'attente à l'autre extrémité du canal prend en charge les listes de distribution. Ce dernier gestionnaire de files d'attente (appelé gestionnaire de files d'attente *de partenariat*) est celui qui reçoit ensuite le message, une fois qu'il a été supprimé de la file d'attente de transmission locale par un agent MCA émetteur.

L'agent MCA émetteur définit l'attribut chaque fois qu'il établit une connexion à l'agent MCA récepteur sur le gestionnaire de files d'attente partenaire. Ainsi, l'agent MCA émetteur peut amener le gestionnaire de files d'attente local à placer dans la file d'attente de transmission uniquement les messages que le gestionnaire de files d'attente partenaire peut traiter correctement.

Cet attribut est principalement utilisé avec les files d'attente de transmission, mais le traitement décrit est effectué quelle que soit l'utilisation définie pour la file d'attente (voir [Attribut d'utilisation](#)).

La valeur est l'une des suivantes :

### **MQDL\_SUPPORTED**

Les messages de liste de distribution peuvent être stockés dans la file d'attente et transmis au gestionnaire de files d'attente partenaire sous cette forme. Cela réduit la quantité de traitement requise pour envoyer le message à plusieurs destinations.

### **MQDL\_NOT\_SUPPORTED**

Les messages de liste de distribution ne peuvent pas être stockés dans la file d'attente car le gestionnaire de files d'attente partenaire ne prend pas en charge les listes de distribution. Si une

application insère un message de liste de distribution et que ce message doit être placé dans cette file d'attente, le gestionnaire de files d'attente fractionne le message de liste de distribution et place les messages individuels dans la file d'attente à la place. Cela augmente la quantité de traitement requise pour envoyer le message à plusieurs destinations, mais garantit que les messages sont traités correctement par le gestionnaire de files d'attente partenaire.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_DIST\_LISTS avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

Cet attribut n'est pas pris en charge sous z/OS.

### **HardenGetAnnulation (MQLONG)**

Pour chaque message, le nombre de fois où le message est extrait par un appel MQGET au sein d'une unité de travail est comptabilisé et cette unité de travail est ensuite annulée.

Tableau 587. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Ce nombre est disponible dans la zone *BackoutCount* du descripteur de message une fois l'appel MQGET terminé.

Le nombre d'annulations de message survit aux redémarrages du gestionnaire de files d'attente. Toutefois, pour s'assurer que le nombre est exact, les informations doivent être *renforcées* (enregistrées sur le disque ou sur une autre unité de stockage permanente) chaque fois qu'un appel MQGET extrait un message dans une unité de travail pour cette file d'attente. Si cette opération n'est pas effectuée, le gestionnaire de files d'attente échoue et les appels MQGET sont annulés, le nombre peut être incrémenté ou non.

Toutefois, les informations de renforcement pour chaque appel MQGET dans une unité de travail imposent des coûts de traitement supplémentaires. Par conséquent, définissez l'attribut **HardenGetBackout** sur MQQA\_BACKOUT\_HARDENED uniquement s'il est essentiel que le nombre soit exact.

Sous *Multiplateformes*, le nombre d'annulations de message est toujours renforcé, quelle que soit la valeur de cet attribut.

Les valeurs suivantes sont possibles :

#### **MQQA\_BACKOUT\_HARDENED**

Le renforcement est utilisé pour s'assurer que le nombre d'annulations pour les messages de cette file d'attente est exact.

#### **MQQA\_BACKOUT\_NOT\_HARDENED**

Le renforcement n'est pas utilisé pour s'assurer que le nombre d'annulations pour les messages de cette file d'attente est correct. Le nombre peut donc être inférieur à ce qu'il devrait être.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_HARDEN\_GET\_BACKOUT avec l'appel MQINQ.

### **IndexType (MQLONG)**

Indique le type d'index que le gestionnaire de files d'attente gère pour les messages de la file d'attente.

Tableau 588. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Le type d'index requis dépend de la manière dont l'application extrait les messages et du fait que la file d'attente est une file d'attente partagée ou une file d'attente non partagée (voir [Attribut QSGDisp](#)). Les valeurs suivantes sont possibles pour *IndexType*:

## **MQIT\_AUCUN**

Aucun index n'est géré par le gestionnaire de files d'attente pour cette file d'attente. Utilisez cette valeur pour les files d'attente qui sont généralement traitées séquentiellement, c'est-à-dire sans utiliser de critères de sélection dans l'appel MQGET.

## **ID\_MSG\_MQ**

Le gestionnaire de files d'attente gère un index qui utilise les identificateurs des messages de la file d'attente. Utilisez cette valeur pour les files d'attente dans lesquelles l'application extrait généralement les messages en utilisant l'identificateur de message comme critère de sélection dans l'appel MQGET.

## **ID\_CORREL\_MQ**

Le gestionnaire de files d'attente gère un index qui utilise les identificateurs de corrélation des messages de la file d'attente. Utilisez cette valeur pour les files d'attente dans lesquelles l'application extrait généralement les messages en utilisant l'identificateur de corrélation comme critère de sélection dans l'appel MQGET.

## **JETON\_MSG\_MQIT\_TOKEN**

**Important :** Ce type d'index ne doit être utilisé que pour les files d'attente utilisées avec le produit IBM MQ Workflow for z/OS .

Le gestionnaire de files d'attente gère un index qui utilise les jetons de message des messages de la file d'attente pour une utilisation avec les fonctions du gestionnaire de charge de travail (WLM) de z/OS.

Vous *devez* spécifier cette option pour les files d'attente gérées par WLM ; ne la spécifiez pour aucun autre type de file d'attente. En outre, n'utilisez pas cette valeur pour une file d'attente dans laquelle une application n'utilise pas les fonctions du gestionnaire de charge de travail z/OS , mais extrait des messages à l'aide du jeton de message comme critère de sélection dans l'appel MQGET.

## **ID\_GROUPE\_MQIT**

Le gestionnaire de files d'attente gère un index qui utilise les identificateurs de groupe des messages de la file d'attente. Cette valeur doit être utilisée pour les files d'attente dans lesquelles l'application extrait des messages à l'aide de l'option MQGMO\_LOGICAL\_ORDER sur l'appel MQGET.

Une file d'attente avec ce type d'index ne peut pas être une file d'attente de transmission. Une file d'attente partagée avec ce type d'index doit être définie pour être mappée à un objet CFSTRUCT de niveau CFLEVEL (3) ou supérieur.

### **Remarque :**

1. L'ordre physique des messages dans une file d'attente avec le type d'index MQIT\_GROUP\_ID n'est pas défini, car la file d'attente est optimisée pour une extraction efficace des messages à l'aide de l'option MQGMO\_LOGICAL\_ORDER sur l'appel MQGET. Cela signifie que l'ordre physique des messages n'est généralement pas l'ordre dans lequel les messages sont arrivés dans la file d'attente.
2. Si une file d'attente MQIT\_GROUP\_ID a une valeur *MsgDeliverySequence* de MQMDS\_PRIORITY, le gestionnaire de files d'attente utilise les priorités de message 0 et 1 pour optimiser l'extraction des messages dans l'ordre logique. Par conséquent, le premier message d'un groupe ne doit pas avoir une priorité égale à zéro ou à un ; si tel est le cas, le message est traité comme s'il avait une priorité égale à deux. La zone *Priority* de la structure MQMD n'est pas modifiée.

Pour plus d'informations sur les groupes de messages, voir la description des options de groupe et de segment dans la zone [MQGMO-Options](#).

Le type d'index qui doit être utilisé dans divers cas est indiqué dans [Tableau 589](#), à la page 893 et [Tableau 590](#), à la page 893.

Tableau 589. Valeurs suggérées ou requises pour le type d'index de file d'attente lorsque MQGMO\_LOGICAL\_ORDER n'est pas spécifié

Critères de sélection lors de l'appel MQGET	Type d'index pour la file d'attente non partagée	Type d'index pour la file d'attente partagée
Aucun	Tous	Tous
<b>Sélection à l'aide d'un identificateur :</b>		
Identificateur de message	MQIT_MSG_ID suggéré	MQIT_NONE ou MQIT_MSG_ID requis ; MQIT_MSG_ID suggéré
Identificateur de corrélation	MQIT_CORREL_ID suggéré	MQIT_CORREL_ID requis
Identificateur de groupe	MQIT_GROUP_ID suggéré	MQIT_GROUP_ID requis
<b>Sélection à l'aide de deux identificateurs :</b>		
Identificateur de message plus identificateur de corrélation	MQIT_MSG_ID ou MQIT_CORREL_ID suggéré	MQIT_NONE ou MQIT_MSG_ID ou MQIT_CORREL_ID requis (Pour plus d'efficacité, il est suggéré que le type d'index soit choisi pour correspondre à la zone MQMD qui aura les clés les plus distinctes)
Identificateur de message plus identificateur de groupe	MQIT_MSG_ID ou MQIT_GROUP_ID suggéré	Non pris en charge
Identificateur de corrélation plus identificateur de groupe	MQIT_CORREL_ID ou MQIT_GROUP_ID suggéré	Non pris en charge
<b>Sélection à l'aide de trois identificateurs :</b>		
Identificateur de message plus identificateur de corrélation plus identificateur de groupe	MQIT_MSG_ID, MQIT_CORREL_ID ou MQIT_GROUP_ID suggéré	Non pris en charge
<b>Sélection à l'aide de critères liés aux groupes :</b>		
Identificateur de groupe plus numéro de séquence de message	MQIT_GROUP_ID requis	MQIT_GROUP_ID requis
Numéro de séquence du message (doit être 1)	MQIT_GROUP_ID requis	MQIT_GROUP_ID requis
<b>Sélection à l'aide d'un jeton de message :</b>		
Jeton de message à utiliser par l'application	Ne pas utiliser MQIT_MSG_TOKEN	
Jeton de message pour l'utilisation de WLM	MQIT_MSG_TOKEN requis	Non pris en charge


Tableau 590. Valeurs suggérées ou requises pour le type d'index de file d'attente lorsque MQGMO\_LOGICAL\_ORDER est spécifié

Critères de sélection lors de l'appel MQGET	Type d'index pour la file d'attente non partagée	Type d'index pour la file d'attente partagée
Aucun	MQIT_GROUP_ID requis	MQIT_GROUP_ID requis

Tableau 590. Valeurs suggérées ou requises pour le type d'index de file d'attente lorsque MQGMO\_LOGICAL\_ORDER est spécifié (suite)

Critères de sélection lors de l'appel MQGET	Type d'index pour la file d'attente non partagée	Type d'index pour la file d'attente partagée
<b>Sélection à l'aide d'un identificateur :</b>		
Identificateur de message	MQIT_GROUP_ID requis	Non pris en charge
Identificateur de corrélation	MQIT_GROUP_ID requis	Non pris en charge
Identificateur de groupe	MQIT_GROUP_ID requis	MQIT_GROUP_ID requis
<b>Sélection à l'aide de deux identificateurs :</b>		
Identificateur de message plus identificateur de corrélation	MQIT_GROUP_ID requis	Non pris en charge
Identificateur de message plus identificateur de groupe	MQIT_GROUP_ID requis	Non pris en charge
Identificateur de corrélation plus identificateur de groupe	MQIT_GROUP_ID requis	Non pris en charge
<b>Sélection à l'aide de trois identificateurs :</b>		
Identificateur de message plus identificateur de corrélation plus identificateur de groupe	MQIT_GROUP_ID requis	Non pris en charge

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_INDEX\_TYPE avec l'appel MQINQ.

 Cet attribut est pris en charge uniquement sous z/OS.

### **InhibitGet (MQLONG)**

Cette option contrôle si les opérations d'extraction de cette file d'attente sont autorisées.

Tableau 591. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X	X	X		

Si la file d'attente est une file d'attente alias, les opérations d'extraction doivent être autorisées pour l'alias et la file d'attente de base au moment de l'opération d'extraction, pour que l'appel MQGET aboutisse. La valeur est l'une des suivantes :

#### **MQQA\_GET\_INHIBÉE**

Les opérations d'extraction sont interdites.

Les appels MQGET échouent avec le code anomalie MQRC\_GET\_INHIBÉ. Cela inclut les appels MQGET qui spécifient MQGMO\_BROWSE\_FIRST ou MQGMO\_BROWSE\_NEXT.

**Remarque :** Si un appel MQGET fonctionnant dans une unité de travail aboutit, la modification de la valeur de l'attribut **InhibitGet** par la suite en MQQA\_GET\_INHIBÉE n'empêche pas la validation de l'unité de travail.

#### **MQQA\_GET\_ALLOWED**

Les opérations d'extraction sont autorisées.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_INHIBIT\_GET avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

### **InhibitPut (MQLONG)**

Indique si les opérations d'insertion pour cette file d'attente sont autorisées.

Tableau 592. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X	X	X	X

S'il existe plusieurs définitions dans le chemin de résolution de nom de file d'attente, les opérations d'insertion doivent être autorisées pour *chaque* définition du chemin (y compris les définitions d'alias de gestionnaire de files d'attente) au moment de l'opération d'insertion, pour que l'appel MQPUT ou MQPUT1 aboutisse. La valeur est l'une des suivantes :

#### **MQQA\_PUT\_INHIBÉ**

Les opérations d'insertion sont interdites.

Les appels MQPUT et MQPUT1 échouent avec le code anomalie MQRC\_PUT\_INHIBÉ.

**Remarque :** Si un appel MQPUT fonctionnant dans une unité de travail aboutit, la modification de la valeur de l'attribut **InhibitPut** par la suite en MQQA\_PUT\_INHIBÉE n'empêche pas la validation de l'unité de travail.

#### **MQQA\_PUT\_ALLOWED**

Les opérations d'insertion sont autorisées.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_INHIBIT\_PUT avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

### **InitiationQName (MQCHAR48)**

Il s'agit du nom d'une file d'attente définie sur le gestionnaire de files d'attente local ; la file d'attente doit être de type MQQT\_LOCAL.

Tableau 593. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X				

Le gestionnaire de files d'attente envoie un message de déclenchement à la file d'attente d'initialisation lorsque le démarrage de l'application est requis suite à l'arrivée d'un message dans la file d'attente à laquelle appartient cet attribut. La file d'attente d'initialisation doit être surveillée par une application de moniteur de déclenchement qui démarre l'application appropriée après la réception du message de déclenchement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_INITIATION\_Q\_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_Q\_NAME\_LENGTH.

### **Longueur MaxMsg(MQLONG)**

Il s'agit d'une limite supérieure pour la longueur du message *physique* le plus long pouvant être placé dans la file d'attente.

Tableau 594. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Toutefois, comme l'attribut de file d'attente **MaxMsgLength** peut être défini indépendamment de l'attribut de gestionnaire de files d'attente **MaxMsgLength**, la limite supérieure réelle de la longueur du message physique le plus long pouvant être placé dans la file d'attente est la plus faible de ces deux valeurs.

Si le gestionnaire de files d'attente prend en charge la segmentation, il est possible qu'une application place un message *logique* plus long que le moindre des deux attributs **MaxMsgLength** , mais uniquement si l'application spécifie l'indicateur MQMF\_SEGMENTATION\_ALLOWED dans MQMD. Si cet indicateur est spécifié, la limite supérieure de la longueur d'un message logique est de 999 999 999 999 octets, mais les contraintes de ressources imposées par le système d'exploitation ou par l'environnement dans lequel l'application est exécutée entraînent une limite inférieure.

Une tentative de placement dans la file d'attente d'un message trop long échoue avec l'un des codes anomalie suivants:

- MQRC\_MSG\_TOO\_BIG\_FOR\_Q si le message est trop grand pour la file d'attente
- MQRC\_MSG\_TOO\_BIG\_FOR\_Q\_MGR si le message est trop grand pour le gestionnaire de files d'attente, mais pas trop grand pour la file d'attente

La limite inférieure de l'attribut **MaxMsgLength** est zéro ; la limite supérieure est de 100 Mo (104 857 600 octets).

Pour plus d'informations, voir le paramètre [MQPUT- BufferLength](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_MAX\_MSG\_LENGTH avec l'appel MQINQ.

### **MaxQDepth (MQLONG)**

Il s'agit de la limite supérieure définie pour le nombre de messages physiques pouvant exister simultanément dans la file d'attente.

*Tableau 595. Types de file d'attente auxquels cet attribut s'applique*

Locale	Modèle	Alias	Eloigné	Cluster
X	X			


Une tentative d'insertion d'un message dans une file d'attente qui contient déjà des messages **MaxQDepth** échoue avec le code anomalie MQRC\_Q\_FULL.

Le traitement des unités de travail et la segmentation des messages peuvent entraîner un nombre réel de messages physiques dans la file d'attente supérieur à **MaxQDepth**. Toutefois, cela n'affecte pas l'extrahibilité du message car tous les messages de la file d'attente peuvent être extraits à l'aide de l'appel MQGET.

La valeur de cet attribut est supérieure ou égale à zéro. La limite supérieure est déterminée par l'environnement:

- Sur les plateformes suivantes, la valeur ne peut pas dépasser 999 999 999:

-  AIX
-  Linux
-  Windows
-  z/OS

-  Sous IBM i, la valeur ne peut pas dépasser 640 000.

**Remarque :** L'espace de stockage disponible pour la file d'attente peut être épuisé même s'il y a moins de messages **MaxQDepth** dans la file d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_MAX\_Q\_DEPTH avec l'appel MQINQ.

### **Séquence MsgDelivery(MQLONG)**



Tableau 596. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Détermine l'ordre dans lequel l'appel MQGET renvoie des messages à l'application:

#### **MQMDS\_FIFO**

Les messages sont renvoyés dans l'ordre FIFO (premier entré, premier sorti).

Un appel MQGET renvoie le *premier* message qui répond aux critères de sélection spécifiés sur l'appel, quelle que soit la priorité du message.

#### **MQMDS\_PRIORITE**

Les messages sont renvoyés par ordre de priorité.

Un appel MQGET renvoie le message *de priorité la plus élevée* qui répond aux critères de sélection spécifiés dans l'appel. Dans chaque niveau de priorité, les messages sont renvoyés dans l'ordre FIFO (premier entré, premier sorti).

- Sous z/OS, si la file d'attente a un *IndexType* de MQIT\_GROUP\_ID, l'attribut **MsgDeliverySequence** indique l'ordre dans lequel les groupes de messages sont renvoyés à l'application. La séquence particulière dans laquelle les groupes sont renvoyés est déterminée par la position ou la priorité du premier message dans chaque groupe. L'ordre physique des messages dans la file d'attente n'est pas défini, car la file d'attente est optimisée pour l'extraction efficace des messages à l'aide de l'option MQGMO\_LOGICAL\_ORDER sur l'appel MQGET.
- Sous z/OS, si *IndexType* est MQIT\_GROUP\_ID et *MsgDeliverySequence* est MQMDS\_PRIORITY, le gestionnaire de files d'attente utilise les priorités de message zéro et un pour optimiser l'extraction des messages dans l'ordre logique. Par conséquent, le premier message d'un groupe ne doit pas avoir une priorité égale à zéro ou à un ; si tel est le cas, le message est traité comme s'il avait une priorité égale à deux. La zone *Priority* de la structure MQMD n'est pas modifiée.

Si les attributs appropriés sont modifiés alors que la file d'attente contient des messages, la séquence de distribution est la suivante:

- L'ordre dans lequel les messages sont renvoyés par l'appel MQGET est déterminé par les valeurs des attributs **MsgDeliverySequence** et **DefPriority** en vigueur pour la file d'attente au moment où le message arrive dans la file d'attente:
  - Si *MsgDeliverySequence* est MQMDS\_FIFO à l'arrivée du message, le message est placé dans la file d'attente comme si sa priorité était *DefPriority*. Cela n'affecte pas la valeur de la zone *Priority* dans le descripteur de message du message ; cette zone conserve la valeur qu'elle avait lors de la première insertion du message.
  - Si *MsgDeliverySequence* est MQMDS\_PRIORITY à l'arrivée du message, le message est placé dans la file d'attente à l'emplacement approprié à la priorité donnée par la zone *Priority* dans le descripteur de message.

Si la valeur de l'attribut **MsgDeliverySequence** est modifiée alors qu'il y a des messages dans la file d'attente, l'ordre des messages dans la file d'attente n'est pas modifié.

Si la valeur de l'attribut **DefPriority** est modifiée alors qu'il y a des messages dans la file d'attente, les messages ne sont pas nécessairement distribués dans l'ordre FIFO, même si l'attribut **MsgDeliverySequence** est défini sur MQMDS\_FIFO; ceux qui ont été placés dans la file d'attente à la priorité la plus élevée sont distribués en premier.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_MSG\_DELIVERY\_SEQUENCE avec l'appel MQINQ.

#### **NonPersistentMessageClass (MQLONG)**

Objectif de fiabilité pour les messages non persistants.

Tableau 597. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Indique les circonstances dans lesquelles les messages non persistants placés dans cette file d'attente sont supprimés:

#### **MQNPM\_CLASS\_NORMAL**

Les messages non persistants sont limités à la durée de vie de la session du gestionnaire de files d'attente ; ils sont supprimés en cas de redémarrage du gestionnaire de files d'attente. Cette valeur est valide uniquement pour les files d'attente non partagées et correspond à la valeur par défaut.

#### **MQNPM\_CLASS\_HIGH**

Le gestionnaire de files d'attente tente de conserver les messages non persistants pendant la durée de vie de la file d'attente. Des messages non persistants peuvent encore être perdus en cas d'échec. Cette valeur est appliquée aux files d'attente partagées.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_NPM\_CLASS avec l'appel MQINQ.

#### **Nombre OpenInput(MQLONG)**

Nombre de descripteurs actuellement valides pour supprimer des messages de la file d'attente à l'aide de l'appel MQGET.

Tableau 598. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X				

Il s'agit du nombre total de descripteurs connus du gestionnaire de files d'attente *local* . Si la file d'attente est une file d'attente partagée, le nombre n'inclut pas les ouvertures pour entrée qui ont été effectuées pour la file d'attente dans d'autres gestionnaires de files d'attente du groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local.

Le nombre inclut les descripteurs dans lesquels une file d'attente alias qui se résout en cette file d'attente a été ouverte en entrée. Le nombre n'inclut pas les descripteurs où la file d'attente a été ouverte pour les actions qui n'incluent pas d'entrée (par exemple, une file d'attente ouverte uniquement pour l'exploration).

La valeur de cet attribut fluctue au fur et à mesure que le gestionnaire de files d'attente fonctionne.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_OPEN\_INPUT\_COUNT avec l'appel MQINQ.

#### **Nombre OpenOutput(MQLONG)**

Il s'agit du nombre de descripteurs actuellement valides pour ajouter des messages à la file d'attente au moyen de l'appel MQPUT.

Tableau 599. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X				

Il s'agit du nombre total de descripteurs connus du gestionnaire de files d'attente *local* ; il n'inclut pas les ouvertures pour les sorties qui ont été effectuées pour cette file d'attente sur les gestionnaires de files d'attente éloignées. Si la file d'attente est une file d'attente partagée, le nombre n'inclut pas les ouvertures pour sortie qui ont été effectuées pour la file d'attente sur d'autres gestionnaires de files d'attente du groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local.

Le nombre inclut les descripteurs dans lesquels une file d'attente alias qui se résout en cette file d'attente a été ouverte pour la sortie. Le nombre n'inclut pas les descripteurs où la file d'attente a été ouverte

pour les actions qui n'incluent pas de sortie (par exemple, une file d'attente ouverte uniquement pour l'interrogation).

La valeur de cet attribut fluctue au fur et à mesure que le gestionnaire de files d'attente fonctionne.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_OPEN\_OUTPUT\_COUNT avec l'appel MQINQ.

### **ProcessName (MQCHAR48)**

Il s'agit du nom d'un objet de processus défini sur le gestionnaire de files d'attente local. L'objet processus identifie un programme qui peut traiter la file d'attente.

<i>Tableau 600. Types de file d'attente auxquels cet attribut s'applique</i>				
<b>Locale</b>	<b>Modèle</b>	<b>Alias</b>	<b>Eloigné</b>	<b>Cluster</b>
X	X			

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_PROCESS\_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_PROCESS\_NAME\_LENGTH.

### **PropertyControl (MQLONG)**

Indique comment les propriétés de message sont traitées pour les messages extraits des files d'attente à l'aide de l'appel MQGET avec l'option MQGMO\_PROPERTIES\_AS\_Q\_DEF.

<i>Tableau 601. Types de file d'attente auxquels cet attribut s'applique</i>				
<b>Locale</b>	<b>Modèle</b>	<b>Alias</b>	<b>Eloigné</b>	<b>Cluster</b>
X	X	X		

La valeur est l'une des suivantes :

#### **MQPROP\_ALL**

Toutes les propriétés du message sont incluses avec le message lorsqu'il est distribué à l'application. Les propriétés, à l'exception de celles du descripteur de message (ou extension), sont placées dans un ou plusieurs en-têtes MQRFH2 dans les données du message. Si un descripteur de message est fourni, le comportement consiste à renvoyer les propriétés dans le descripteur de message.

#### **COMPATIBILITE\_MQPROP\_COMPATIBILITÉ**

Si le message contient une propriété avec un préfixe mcd., jms., usr. ou mqext., Toutes les propriétés de message sont distribuées à l'application dans un en-tête MQRFH2. Sinon, toutes les propriétés du message, à l'exception de celles du descripteur de message (ou extension), sont supprimées et ne sont plus accessibles à l'application. Il s'agit de la valeur par défaut ; elle permet aux applications qui s'attendent à ce que les propriétés associées à JMS soient dans un en-tête MQRFH2 dans les données de message de continuer à fonctionner sans modification. Si un descripteur de message est fourni, le comportement consiste à renvoyer les propriétés dans le descripteur de message.

#### **MQPROP\_FORCE\_MQRFH2**

Les propriétés sont toujours renvoyées dans les données du message dans un en-tête MQRFH2, indépendamment du fait que l'application indique un descripteur de message. Un descripteur de message valide fourni dans la zone MsgHandle de la structure MQGMO sur l'appel MQGET est ignoré. Les propriétés du message ne sont pas accessibles via l'identificateur de message.

#### **MQPROP\_NONE**

Toutes les propriétés du message, à l'exception de celles du descripteur de message (ou de l'extension), sont supprimées du message avant que celui-ci ne soit distribué à l'application. Si un descripteur de message est fourni, le comportement consiste à renvoyer les propriétés dans le descripteur de message.

Ce paramètre est applicable aux files d'attente locales, d'alias et de modèle. Pour déterminer sa valeur, utilisez le sélecteur MQIA\_PROPERTY\_CONTROL avec l'appel MQINQ.

### **Événement QDepthHigh(MQLONG)**

Indique si les événements Longueur élevée de la file d'attente sont générés.

Tableau 602. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Un événement Longueur élevée de file d'attente indique qu'une application a inséré un message dans une file d'attente, ce qui a entraîné un nombre de messages dans la file d'attente supérieur ou égal au seuil haut de longueur de file d'attente (voir l'attribut **QDepthHighLimit**).

**Remarque :** La valeur de cet attribut peut changer dynamiquement.

La valeur est l'une des suivantes :

#### **MQEVR\_DISABLED**

Génération de rapports d'événements désactivée.

#### **MQEVR\_ENABLED**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_Q\_DEPTH\_HIGH\_EVENT avec l'appel MQINQ.

Cet attribut est pris en charge sous z/OS, mais l'appel MQINQ ne peut pas être utilisé pour déterminer sa valeur.

### **Limite QDepthHigh(MQLONG)**

Il s'agit du seuil auquel le nombre de lignes de la file d'attente est comparé pour générer un événement Longueur élevée de la file d'attente.

Tableau 603. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Cet événement indique qu'une application a inséré un message dans une file d'attente, ce qui a entraîné un nombre de messages dans la file d'attente supérieur ou égal au seuil maximal de longueur de la file d'attente. Voir [Attribut d'événementQDepthHigh](#).

La valeur est exprimée en pourcentage de la longueur maximale de la file d'attente (attribut **MaxQDepth**) et est supérieure ou égale à 0 et inférieure ou égale à 100. La valeur par défaut est 80.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_Q\_DEPTH\_HIGH\_LIMIT avec l'appel MQINQ.

Cet attribut est pris en charge sous z/OS, mais l'appel MQINQ ne peut pas être utilisé pour déterminer sa valeur.

### **Événement QDepthLow(MQLONG)**

Cette option contrôle si les événements Longueur faible de la file d'attente sont générés.

Tableau 604. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Un événement Longueur basse de file d'attente indique qu'une application a extrait un message d'une file d'attente et que le nombre de messages de la file d'attente est devenu inférieur ou égal au seuil bas de longueur de la file d'attente (voir [QDepthLow](#) ).

**Remarque :** La valeur de cet attribut peut changer dynamiquement.

La valeur est l'une des suivantes :

**MQEVR\_DISABLED**

Génération de rapports d'événements désactivée.

**MQEVR\_ENABLED**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_Q\_DEPTH\_LOW\_EVENT avec l'appel MQINQ.

Cet attribut est pris en charge sous z/OS, mais l'appel MQINQ ne peut pas être utilisé pour déterminer sa valeur.

**Limite QDepthLow(MQLONG)**

Il s'agit du seuil auquel le nombre de lignes de la file d'attente est comparé pour générer un événement Longueur faible de la file d'attente.

<i>Tableau 605. Types de file d'attente auxquels cet attribut s'applique</i>				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Cet événement indique qu'une application a extrait un message d'une file d'attente et que le nombre de messages de la file d'attente est inférieur ou égal au seuil bas de longueur de la file d'attente. Voir [Attribut d'événementQDepthLow](#).

La valeur est exprimée en pourcentage de la longueur maximale de la file d'attente (attribut**MaxQDepth** ) et est supérieure ou égale à 0 et inférieure ou égale à 100. La valeur par défaut est 20.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_Q\_DEPTH\_LOW\_LIMIT avec l'appel MQINQ.

Cet attribut est pris en charge sous z/OS, mais l'appel MQINQ ne peut pas être utilisé pour déterminer sa valeur.

**Événement QDepthMax(MQLONG)**

Cette option contrôle si les événements File d'attente saturée sont générés. Un événement File d'attente saturée indique qu'une insertion dans une file d'attente a été rejetée car la file d'attente est saturée, c'est-à-dire que la longueur de la file d'attente a déjà atteint sa valeur maximale.

<i>Tableau 606. Types de file d'attente auxquels cet attribut s'applique</i>				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

**Remarque :** La valeur de cet attribut peut changer dynamiquement.

La valeur est l'une des suivantes :

**MQEVR\_DISABLED**

Génération de rapports d'événements désactivée.

**MQEVR\_ENABLED**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_Q\_DEPTH\_MAX\_EVENT avec l'appel MQINQ.

Cet attribut est pris en charge sous z/OS, mais l'appel MQINQ ne peut pas être utilisé pour déterminer sa valeur.

### **QDesc (MQCHAR64)**

Utilisez cette zone pour les commentaires descriptifs.

<i>Tableau 607. Types de file d'attente auxquels cet attribut s'applique</i>				
<b>Locale</b>	<b>Modèle</b>	<b>Alias</b>	<b>Eloigné</b>	<b>Cluster</b>
X	X	X	X	X

Le contenu de la zone n'est pas significatif pour le gestionnaire de files d'attente, mais ce dernier peut exiger que la zone ne contienne que des caractères pouvant être affichés. Elle ne peut pas contenir de caractères nuls ; si nécessaire, elle est remplie à droite avec des blancs. Dans une installation DBCS, la zone peut contenir des caractères DBCS (avec une longueur de zone maximale de 64 octets).

**Remarque :** Si cette zone contient des caractères qui ne figurent pas dans le jeu de caractères du gestionnaire de files d'attente (tel que défini par l'attribut de gestionnaire de files d'attente **CodedCharSetId**), ces caractères peuvent être convertis de manière incorrecte si cette zone est envoyée à un autre gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_Q\_DESC avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_Q\_DESC\_LENGTH.

### **QName (MQCHAR48)**

Il s'agit du nom d'une file d'attente définie sur le gestionnaire de files d'attente local.

<i>Tableau 608. Types de file d'attente auxquels cet attribut s'applique</i>				
<b>Locale</b>	<b>Modèle</b>	<b>Alias</b>	<b>Eloigné</b>	<b>Cluster</b>
X		X	X	X

Toutes les files d'attente définies sur un gestionnaire de files d'attente partagent le même espace de nom de file d'attente. Par conséquent, une file d'attente MQQT\_LOCAL et une file d'attente MQQT\_ALIAS ne peuvent pas avoir le même nom.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_Q\_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_Q\_NAME\_LENGTH.

### **QServiceInterval (MQLONG)**

Il s'agit de l'intervalle de service utilisé à des fins de comparaison pour générer des événements d'intervalle de service élevé et d'intervalle de service OK.

<i>Tableau 609. Types de file d'attente auxquels cet attribut s'applique</i>				
<b>Locale</b>	<b>Modèle</b>	<b>Alias</b>	<b>Eloigné</b>	<b>Cluster</b>
X	X			

Voir [Attribut d'événementQServiceInterval](#).

La valeur est exprimée en millisecondes et est supérieure ou égale à zéro et inférieure ou égale à 999 999 999.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_Q\_SERVICE\_INTERVAL avec l'appel MQINQ.

Cet attribut est pris en charge sous z/OS, mais l'appel MQINQ ne peut pas être utilisé pour déterminer sa valeur.

### **Événement QServiceInterval(MQLONG)**

Indique si les événements Intervalle de service élevé ou Intervalle de service OK sont générés.

<i>Tableau 610. Types de file d'attente auxquels cet attribut s'applique</i>				
<b>Locale</b>	<b>Modèle</b>	<b>Alias</b>	<b>Eloigné</b>	<b>Cluster</b>
X	X			

- Un événement Intervalle de service élevé est généré lorsqu'une vérification indique qu'aucun message n'a été extrait de la file d'attente pendant au moins le temps indiqué par l'attribut **QServiceInterval**.
- Un événement d'intervalle de service OK est généré lorsqu'une vérification indique que des messages ont été extraits de la file d'attente dans le délai indiqué par l'attribut **QServiceInterval**.

**Remarque :** La valeur de cet attribut peut changer dynamiquement.

La valeur est l'une des suivantes :

#### **MQQSIE\_HIGH**

Événements d'intervalle élevé du service de file d'attente activés.

- Les événements d'intervalle de service de file d'attente élevé sont **activés** et
- Les événements d'intervalle de service de file d'attente OK sont **désactivés**.

#### **MQQSIE\_OK**

Événements d'intervalle de service de file d'attente OK activés.

- Les événements d'intervalle de service de file d'attente élevé sont **désactivés** et
- Les événements d'intervalle de service de file d'attente OK sont **activés**.

#### **MQQSIE\_NONE**

Aucun événement d'intervalle de service de file d'attente n'est activé.

- Les événements d'intervalle de service de file d'attente élevé sont **désactivés** et
- Les événements d'intervalle de service de file d'attente OK sont également **désactivés**.

Pour les files d'attente partagées, la valeur de cet attribut est ignorée ; la valeur MQQSIE\_NONE est prise en compte.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_Q\_SERVICE\_INTERVAL\_EVENT avec l'appel MQINQ.

Sous z/OS, vous ne pouvez pas utiliser l'appel MQINQ pour déterminer la valeur de cet attribut.

### **QSGDisp (MQLONG)**

Indique la disposition de la file d'attente.

<i>Tableau 611. Types de file d'attente auxquels cet attribut s'applique</i>				
<b>Locale</b>	<b>Modèle</b>	<b>Alias</b>	<b>Eloigné</b>	<b>Cluster</b>
X		X	X	

La valeur est l'une des suivantes :

#### **MQQSGD\_Q\_DIR**

L'objet possède une disposition de gestionnaire de files d'attente. Cela signifie que la définition d'objet n'est connue que du gestionnaire de files d'attente local ; la définition n'est pas connue des autres gestionnaires de files d'attente du groupe de partage de files d'attente.

Chaque gestionnaire de files d'attente du groupe de partage de files d'attente peut avoir un objet ayant le même nom et le même type que l'objet en cours, mais il s'agit d'objets distincts et il n'y a pas de corrélation entre ces objets. Leurs attributs ne sont pas contraints d'être identiques les uns aux autres.


#### **MQQSGD\_COPY**

L'objet est une copie locale d'une définition d'objet maître qui existe dans le référentiel partagé. Chaque gestionnaire de files d'attente du groupe de partage de files d'attente peut disposer de sa propre copie de l'objet. Au départ, toutes les copies ont les mêmes attributs, mais en utilisant les commandes MQSC, vous pouvez modifier chaque copie de sorte que ses attributs diffèrent de ceux des autres copies. Les attributs des copies sont resynchronisés lorsque la définition principale du référentiel partagé est modifiée.

#### **MQQSGD\_SHARED**

L'objet a une disposition partagée. Cela signifie qu'il existe dans le référentiel partagé une seule instance de l'objet connue de tous les gestionnaires de files d'attente du groupe de partage de files d'attente. Lorsqu'un gestionnaire de files d'attente du groupe accède à l'objet, il accède à l'instance partagée unique de l'objet.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_QSG\_DISP avec l'appel MQINQ.

 Cet attribut est pris en charge uniquement sous z/OS.

### **QueueAccounting (MQLONG)**

*Tableau 612. Types de file d'attente auxquels cet attribut s'applique*

Locale	Modèle	Alias	Eloigné	Cluster
X	X	X	X	

Cette option contrôle la collecte des données comptables pour la file d'attente. Pour que les données de comptabilité soient collectées pour cette file d'attente, les données de comptabilité de cette connexion doivent également être activées, à l'aide de l'attribut QMGR ACCTQ ou de la zone Options de la structure MQCNO sur l'appel MQCONN.

Cet attribut comporte l'une des valeurs suivantes :

#### **MQMON\_Q\_DIR**

Les données comptables de cette file d'attente sont collectées en fonction de la valeur de l'attribut QMGR ACCTQ. Il s'agit du paramètre par défaut.

#### **MQMON\_OFF**

Ne collectez pas de données de comptabilité pour cette file d'attente.

#### **MQMON\_ON**

Collectez les données de comptabilité pour cette file d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_ACCOUNTING\_Q avec l'appel MQINQ.

### **QueueMonitoring (MQLONG)**

Ctrl la collecte des données de surveillance en ligne pour les files d'attente.

*Tableau 613. Types de file d'attente auxquels cet attribut s'applique*

Locale	Modèle	Alias	Eloigné	Cluster
X	X			

La valeur est l'une des suivantes :



**MQMON\_Q\_DIR**

Collectez les données de surveillance en fonction de la valeur de l'attribut de gestionnaire de files d'attente **QueueMonitoring** . Il s'agit de la valeur par défaut.

**MQMON\_OFF**

La collecte des données de surveillance en ligne est désactivée pour cette file d'attente.

**MQMON\_FAIBLE**

Si la valeur de l'attribut de gestionnaire de files d'attente **QueueMonitoring** n'est pas MQMON\_NONE, la collecte de données de surveillance en ligne est activée, avec un faible débit de collecte de données pour cette file d'attente.

**MQMON\_MEDIUM**

Si la valeur de l'attribut de gestionnaire de files d'attente **QueueMonitoring** n'est pas MQMON\_NONE, la collecte de données de surveillance en ligne est activée, avec un taux modéré de collecte de données pour cette file d'attente.

**MQMON\_ELEVE**

Si la valeur de l'attribut de gestionnaire de files d'attente **QueueMonitoring** n'est pas MQMON\_NONE, la collecte de données de surveillance en ligne est activée, avec un taux élevé de collecte de données pour cette file d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_MONITORING\_Q avec l'appel MQINQ.

**QueueStatistics (MQCHAR12)**

<i>Tableau 614. Types de file d'attente auxquels cet attribut s'applique</i>				
Locale	Modèle	Alias	Eloigné	Cluster
X	X	X	X	

Permet de contrôler la collecte des données statistiques pour la file d'attente.

Cet attribut comporte l'une des valeurs suivantes :

**MQMON\_Q\_DIR**

Les données comptables de cette file d'attente sont collectées en fonction de la valeur de l'attribut QMGR STATQ. Il s'agit du paramètre par défaut.

**MQMON\_OFF**

Désactivez la collecte de données statistiques pour cette file d'attente.

**MQMON\_ON**

Activez la collecte des données statistiques pour cette file d'attente.

**Type de file d'attente (MQLONG)**

<i>Tableau 615. Types de file d'attente auxquels cet attribut s'applique</i>				
Locale	Modèle	Alias	Eloigné	Cluster
X		X	X	X

Il s'agit du type de file d'attente ; il possède l'une des valeurs suivantes:

**MQQT\_ALIAS (alias MQ)**

Définition de file d'attente alias.

**MQQT\_CLUSTER**

File d'attente de cluster.

**MQQT\_LOCAL**

File d'attente locale.

## MQQT\_REMOTE

Définition locale d'une file d'attente éloignée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_Q\_TYPE avec l'appel MQINQ.

### RemoteQMgrNom (MQCHAR48)

Tableau 616. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
			X	

Il s'agit du nom du gestionnaire de files d'attente éloignées sur lequel la file d'attente **RemoteQName** est définie. Si la file d'attente **RemoteQName** a la valeur **QSGDisp** MQQSGD\_COPY ou MQQSGD\_SHARED, **RemoteQMgrName** peut être le nom du groupe de partage de files d'attente qui possède **RemoteQName**.

Si une application ouvre la définition locale d'une file d'attente éloignée, **RemoteQMgrName** ne doit pas être vide et ne doit pas être le nom du gestionnaire de files d'attente local. Si **XmitQName** est vide, la file d'attente locale portant le même nom que **RemoteQMgrName** est utilisée comme file d'attente de transmission. S'il n'existe aucune file d'attente portant le nom **RemoteQMgrName**, la file d'attente identifiée par l'attribut de gestionnaire de files d'attente **DefXmitQName** est utilisée.

Si cette définition est utilisée pour un alias de gestionnaire de files d'attente, **RemoteQMgrName** est le nom du gestionnaire de files d'attente dont l'alias est utilisé. Il peut s'agir du nom du gestionnaire de files d'attente local. Sinon, si **XmitQName** est vide lors de l'ouverture, il doit y avoir une file d'attente locale portant le même nom que **RemoteQMgrName**; cette file d'attente est utilisée comme file d'attente de transmission.

Si cette définition est utilisée pour un alias de réponse, il s'agit du nom du gestionnaire de files d'attente qui doit être **ReplyToQMgr**.

**Remarque :** Aucune validation n'est effectuée sur la valeur spécifiée pour cet attribut lorsque la définition de file d'attente est créée ou modifiée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_REMOTE\_Q\_MGR\_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_Q\_MGR\_NAME\_LENGTH.

### RemoteQName (MQCHAR48)

Tableau 617. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
			X	

Il s'agit du nom de la file d'attente telle qu'elle est connue sur le gestionnaire de files d'attente éloignées **RemoteQMgrName**.

Si une application ouvre la définition locale d'une file d'attente éloignée, lorsque l'ouverture a lieu, **RemoteQName** ne doit pas être vide.

Si cette définition est utilisée pour une définition d'alias de gestionnaire de files d'attente, lorsque l'ouverture a lieu, la zone **RemoteQName** doit être vide.

Si la définition est utilisée pour un alias de réponse, ce nom correspond au nom de la file d'attente qui doit être **ReplyToQ**.

**Remarque :** Aucune validation n'est effectuée sur la valeur spécifiée pour cet attribut lorsque la définition de file d'attente est créée ou modifiée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_REMOTE\_Q\_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_Q\_NAME\_LENGTH.

## RetentionInterval (MQLONG)

Il s'agit de la durée pendant laquelle la file d'attente doit être conservée. Une fois ce délai écoulé, la file d'attente peut être supprimée.

Tableau 618. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

La durée est mesurée en heures, à compter de la date et de l'heure de création de la file d'attente. La date et l'heure de création de la file d'attente sont enregistrées dans les attributs **CreationDate** et **CreationTime**.

Ces informations sont fournies pour permettre à une application de nettoyage ou à l'opérateur d'identifier et de supprimer les files d'attente qui ne sont plus nécessaires.

**Remarque :** Le gestionnaire de files d'attente n'effectue aucune action pour supprimer des files d'attente en fonction de cet attribut ou pour empêcher la suppression de files d'attente avec un intervalle de conservation qui n'est pas arrivé à expiration. Il incombe à l'utilisateur d'effectuer les actions requises.

Utilisez un intervalle de conservation réaliste pour empêcher l'accumulation de files d'attente dynamiques permanentes (voir [attributDefinitionType](#)). Toutefois, cet attribut peut également être utilisé avec des files d'attente prédéfinies.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_RETENTION\_INTERVAL avec l'appel MQINQ.

## Portée (MQLONG)

Contrôle si une entrée de cette file d'attente existe également dans un répertoire de cellules.

Tableau 619. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X		X	X	

Un répertoire de cellule est fourni par un service de nom installable. La valeur est l'une des suivantes :

### MQSCO\_Q\_DIR

La définition de file d'attente a la portée du gestionnaire de files d'attente: la définition de la file d'attente ne s'étend pas au-delà du gestionnaire de files d'attente qui la possède. Pour ouvrir la file d'attente à des fins de sortie à partir d'un autre gestionnaire de files d'attente, le nom du gestionnaire de files d'attente propriétaire doit être spécifié ou l'autre gestionnaire de files d'attente doit avoir une définition locale de la file d'attente.

### MQSCO\_CELL

La définition de file d'attente a une portée de type cellule: la définition de file d'attente est également placée dans un répertoire de cellule accessible à tous les gestionnaires de files d'attente de la cellule. La file d'attente peut être ouverte pour la sortie à partir de n'importe quel gestionnaire de files d'attente de la cellule en spécifiant le nom de la file d'attente. Il n'est pas nécessaire de spécifier le nom du gestionnaire de files d'attente propriétaire de la file d'attente. Toutefois, la définition de file d'attente n'est pas disponible pour les gestionnaires de files d'attente de la cellule qui ont également une définition locale d'une file d'attente portant ce nom, car la définition locale est prioritaire.

Un répertoire de cellule est fourni par un service de nom installable.

Les files d'attente modèles et dynamiques ne peuvent pas avoir de portée cellule.

Cette valeur n'est valide que si un service annuaire prenant en charge un répertoire de cellule a été configuré.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_SCOPE avec l'appel MQINQ.

La prise en charge de cet attribut est soumise aux restrictions suivantes:

- Sous IBM i, l'attribut est pris en charge, mais seul MQSCO\_Q\_MGR est valide.
- Sous z/OS, l'attribut n'est pas pris en charge.

### Partageabilité (MQLONG)

Indique si la file d'attente peut être ouverte en entrée plusieurs fois simultanément.

Tableau 620. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

La valeur est l'une des suivantes :

#### MQQA\_SHAREABLE

La file d'attente est partageable.

Plusieurs ouvertures avec l'option MQOO\_INPUT\_SHARED sont autorisées.

#### MQQA\_NOT\_SHAREABLE

La file d'attente n'est pas partageable.

Un appel MQOPEN avec l'option MQOO\_INPUT\_SHARED est traité comme MQOO\_INPUT\_EXCLUSIVE.


Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_SHAREABILITY avec l'appel MQINQ.

### StorageClass (MQCHAR8)

Il s'agit d'un nom défini par l'utilisateur qui définit la mémoire physique utilisée pour contenir la file d'attente. En pratique, un message n'est écrit sur le disque que s'il doit être paginé hors de sa mémoire tampon.

Tableau 621. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_STORAGE\_CLASS avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_STORAGE\_CLASS\_LENGTH.

 Cet attribut est pris en charge uniquement sous z/OS.

### TriggerControl (MQLONG)

Cette option contrôle si les messages de déclenchement sont écrits dans une file d'attente d'initialisation pour démarrer une application afin de traiter la file d'attente.

Tableau 622. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Il s'agit de l'un des éléments suivants:

#### MQTC\_OFF

Aucun message de déclenchement ne doit être écrit pour cette file d'attente. La valeur de *TriggerType* n'est pas pertinente dans ce cas.

#### MQTC\_ON

Les messages de déclenchement doivent être écrits pour cette file d'attente lorsque les événements de déclenchement appropriés se produisent.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_TRIGGER\_CONTROL avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

### **TriggerData (MQCHAR64)**

Il s'agit des données de format libre que le gestionnaire de files d'attente insère dans le message de déclenchement lorsqu'un message arrivant dans cette file d'attente entraîne l'écriture d'un message de déclenchement dans la file d'attente d'initialisation.

Tableau 623. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Le contenu de ces données n'a aucune importance pour le gestionnaire de files d'attente. Il est significatif soit pour l'application du moniteur de déclenchement qui traite la file d'attente d'initialisation, soit pour l'application que le moniteur de déclenchement démarre.

La chaîne de caractères ne doit pas contenir de valeurs nulles. Il est rempli à droite avec des blancs si nécessaire.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_TRIGGER\_DATA avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET. La longueur de cet attribut est donnée par MQ\_TRIGGER\_DATA\_LENGTH.

### **TriggerDepth (MQLONG)**

Tableau 624. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Il s'agit du nombre de messages de priorité *TriggerMsgPriority* ou supérieure qui doivent se trouver dans la file d'attente avant qu'un message de déclenchement ne soit écrit. Cela s'applique lorsque *TriggerType* est défini sur MQTT\_DEPTH. La valeur de *TriggerDepth* est supérieure ou égale à 1. Cet attribut n'est pas utilisé autrement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_TRIGGER\_DEPTH avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

### **Priorité TriggerMsg(MQLONG)**

Il s'agit de la priorité de message au-dessous de laquelle les messages ne contribuent pas à la génération de messages de déclenchement (c'est-à-dire que le gestionnaire de files d'attente ignore ces messages lorsqu'il décide de générer ou non un message de déclenchement).

Tableau 625. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

*TriggerMsgPriority* peut être compris entre zéro (valeur la plus faible) et *MaxPriority* (valeur la plus élevée ; voir [AttributMaxPriority](#)) ; la valeur zéro entraîne la contribution de tous les messages à la génération de messages de déclenchement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_TRIGGER\_MSG\_PRIORITY avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

### **TriggerType (MQLONG)**

Cette option contrôle les conditions dans lesquelles les messages de déclenchement sont écrits suite à l'arrivée de messages dans cette file d'attente.

Tableau 626. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Elle a l'une des valeurs suivantes :

#### **MQTT\_AUCUN**

Aucun message de déclenchement n'est écrit suite à des messages dans cette file d'attente. Cela a le même effet que de définir *TriggerControl* sur MQTC\_OFF.

#### **MQTT\_PREMIER**

Un message de déclenchement est écrit chaque fois que le nombre de messages de priorité *TriggerMsgPriority* ou supérieure dans la file d'attente passe de 0 à 1.

#### **MQTT EVERY**

Un message de déclenchement est écrit chaque fois qu'un message de priorité *TriggerMsgPriority* ou supérieure arrive dans la file d'attente.

#### **MQTT\_DEPTH**

Un message de déclenchement est écrit chaque fois que le nombre de messages de priorité *TriggerMsgPriority* ou supérieure dans la file d'attente est égal ou supérieur à *TriggerDepth*. Une fois que le message de déclenchement a été écrit, *TriggerControl* est défini sur MQTC\_OFF pour empêcher tout déclenchement supplémentaire jusqu'à ce qu'il soit explicitement activé à nouveau.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_TRIGGER\_TYPE avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

### **Syntaxe (MQLONG)**

Indique à quoi sert la file d'attente.

Tableau 627. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X	X			

La valeur est l'une des suivantes :

#### **MQUS\_NORMAL**

Il s'agit d'une file d'attente utilisée par les applications lors de l'insertion et de l'extraction de messages ; la file d'attente n'est pas une file d'attente de transmission.

#### **TRANSMIS\_MQUS\_TRANSMISSION**

Il s'agit d'une file d'attente utilisée pour le stockage des messages destinés aux gestionnaires de files d'attente éloignées. Lorsqu'une application envoie un message à une file d'attente éloignée, le gestionnaire de files d'attente local stocke temporairement le message dans la file d'attente de transmission appropriée dans un format spécial. Un agent de canal de transmission lit ensuite le message à partir de la file d'attente de transmission et le transporte vers le gestionnaire de files d'attente éloignées. Pour plus d'informations sur la configuration de l'administration à distance, voir [Configuration des gestionnaires de files d'attente pour l'administration à distance](#).

Seules les applications privilégiées peuvent ouvrir une file d'attente de transmission pour MQOO\_OUTPUT afin d'y placer directement des messages. Généralement, seules les applications utilitaires le font. Vérifiez que le format des données de message est correct (voir «MQXQH-en-tête de file d'attente de transmission», à la page 647) ou des erreurs peuvent se produire lors du processus de transmission. Le contexte n'est pas transmis ou défini sauf si l'une des options de contexte MQPMO\_\*\_CONTEXT est spécifiée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_USAGE avec l'appel MQINQ.

### **XmitQName (MQCHAR48)**

Il s'agit du nom de la file d'attente de transmission. Si cet attribut n'est pas à blanc lorsqu'une ouverture se produit, soit pour une file d'attente éloignée, soit pour une définition d'alias de gestionnaire de files d'attente, il indique le nom de la file d'attente de transmission locale à utiliser pour le transfert du message.

Tableau 628. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
			X	

Si **XmitQName** est vide, la file d'attente locale dont le nom est identique à **RemoteQMGrName** est utilisée comme file d'attente de transmission. S'il n'existe aucune file d'attente portant le nom **RemoteQMGrName**, la file d'attente identifiée par l'attribut de gestionnaire de files d'attente **DefXmitQName** est utilisée.

Cet attribut est ignoré si la définition est utilisée en tant qu'alias de gestionnaire de files d'attente et que **RemoteQMGrName** est le nom du gestionnaire de files d'attente local. Il est également ignoré lorsque la définition est utilisée comme définition d'alias de file d'attente de réponses.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_XMIT\_Q\_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_Q\_NAME\_LENGTH.

### **Attributs des listes de noms**

Le tableau suivant récapitule les attributs spécifiques aux listes de noms. Les attributs sont décrits par ordre alphabétique.

Les listes de noms sont prises en charge sur tous les systèmes IBM MQ , plus les IBM MQ MQI clients connectés à ces systèmes.

**Remarque :** Les noms des attributs affichés dans cette section sont des noms descriptifs utilisés avec les appels MQINQ et MQSET ; les noms sont identiques à ceux des commandes PCF. Lorsque des commandes MQSC sont utilisées pour définir, modifier ou afficher des attributs, d'autres noms abrégés sont utilisés ; voir [Commandes MQSC](#) pour plus d'informations.

Tableau 629. Attributs des listes de noms	
Attribut	Description
<a href="#">AlterationDate</a>	Date de la dernière modification de la définition
<a href="#">AlterationTime</a>	Heure de la dernière modification de la définition
<a href="#">NameCount</a>	Nombre de noms dans la liste de noms
<a href="#">NamelistDesc</a>	Description de la liste de noms
<a href="#">NamelistName</a>	Nom de la liste de noms
<a href="#">Noms</a>	Liste de noms <i>NameCount</i>
<a href="#">NamelistType</a>	Type de liste de noms
<a href="#">QSGDisp</a>	Disposition d'un groupe de partage de files d'attente

### **AlterationDate (MQCHAR12)**

Il s'agit de la date de la dernière modification de la définition. Le format de la date est YYYY-MM-DD, complété par deux blancs de fin pour que la longueur soit de 12 octets.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_ALTERATION\_DATE avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_DATE\_LENGTH.

### **AlterationTime (MQCHAR8)**

Il s'agit de l'heure à laquelle la définition a été modifiée pour la dernière fois. Le format de l'heure est HH.MM.SS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_ALTERATION\_TIME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_TIME\_LENGTH.

### ***NameCount (MQLONG)***

Il s'agit du nombre de noms figurant dans la liste de noms. Elle est supérieure ou égale à zéro. La valeur suivante est définie:

#### **MQNC\_MAX\_NAMELIST\_NAME\_COUNT**

Nombre maximal de noms dans une liste de noms.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_NAME\_COUNT avec l'appel MQINQ.

### ***NamelistDesc (MQCHAR64)***

Utilisez cette zone pour les commentaires descriptifs ; sa valeur est établie par le processus de définition. Le contenu de la zone n'est pas significatif pour le gestionnaire de files d'attente, mais ce dernier peut exiger que la zone ne contienne que des caractères pouvant être affichés. Elle ne peut pas contenir de caractères nuls ; si nécessaire, elle est remplie à droite avec des blancs. Dans une installation DBCS, cette zone peut contenir des caractères DBCS (avec une longueur de zone maximale de 64 octets).

**Remarque :** Si cette zone contient des caractères qui ne figurent pas dans le jeu de caractères du gestionnaire de files d'attente (tel que défini par l'attribut de gestionnaire de files d'attente **CodedCharSetId**), ces caractères peuvent être convertis de manière incorrecte si cette zone est envoyée à un autre gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_NAMELIST\_DESC avec l'appel MQINQ.

La longueur de cet attribut est donnée par MQ\_NAMELIST\_DESC\_LENGTH.

### ***NamelistName (MQCHAR48)***

Il s'agit du nom d'une liste de noms définie sur le gestionnaire de files d'attente local. Pour plus d'informations sur les noms de liste de noms, voir la section [Autres noms d'objet](#).

Chaque liste de noms a un nom qui est différent des noms des autres listes de noms appartenant au gestionnaire de files d'attente, mais qui peut dupliquer les noms des autres objets de gestionnaire de files d'attente de types différents (par exemple, les files d'attente).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_NAMELIST\_NAME avec l'appel MQINQ.

La longueur de cet attribut est donnée par MQ\_NAMELIST\_NAME\_LENGTH.

### ***NamelistType (MQLONG)***

Indique la nature des noms dans la liste de noms et la manière dont la liste de noms est utilisée. Ses valeurs sont l'une des suivantes :

#### **MQNT\_AUCUN**

Liste de noms sans type affecté.

#### **MQNT\_Q**

Liste de noms contenant les noms des files d'attente.

#### **MQNT\_CLUSTER**

Liste de noms contenant les noms des clusters.

#### **INFO MQNT\_AUTH\_INFO**

Liste de noms contenant les noms des objets d'informations d'authentification.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_NAMELIST\_TYPE avec l'appel MQINQ.

 Cet attribut est pris en charge uniquement sous z/OS.



## Noms (MQCHAR48xNameCount)

Il s'agit d'une liste de noms *NameCount*, où chaque nom correspond au nom d'un objet défini dans le gestionnaire de files d'attente local. Pour plus d'informations sur les noms d'objet, voir [Règles de dénomination des objets IBM MQ](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_NAMES avec l'appel MQINQ.

La longueur de chaque nom dans la liste est indiquée par MQ\_OBJECT\_NAME\_LENGTH.

## QSGDisp (MQLONG)

Indique la disposition de la liste de noms. La valeur est l'une des suivantes :

### MQQSGD\_Q\_DIR


L'objet possède une disposition de gestionnaire de files d'attente: la définition d'objet est connue uniquement du gestionnaire de files d'attente local ; la définition n'est pas connue des autres gestionnaires de files d'attente du groupe de partage de files d'attente.

Chaque gestionnaire de files d'attente du groupe de partage de files d'attente peut avoir un objet ayant le même nom et le même type que l'objet en cours, mais il s'agit d'objets distincts et il n'y a pas de corrélation entre ces objets. Leurs attributs ne sont pas contraints d'être identiques les uns aux autres.

### MQQSGD\_COPY

L'objet est une copie locale d'une définition d'objet maître qui existe dans le référentiel partagé. Chaque gestionnaire de files d'attente du groupe de partage de files d'attente peut disposer de sa propre copie de l'objet. Au départ, toutes les copies ont les mêmes attributs, mais vous pouvez modifier chaque copie à l'aide des commandes MQSC, de sorte que ses attributs diffèrent de ceux des autres copies. Les attributs des copies sont resynchronisés lorsque la définition principale du référentiel partagé est modifiée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_QSG\_DISP avec l'appel MQINQ.

 Cet attribut est pris en charge uniquement sous z/OS.

## Attributs des définitions de processus

Le tableau suivant récapitule les attributs spécifiques aux définitions de processus. Les attributs sont décrits par ordre alphabétique.

**Remarque :** Les noms des attributs de cette section sont des noms descriptifs utilisés avec les appels MQINQ et MQSET ; les noms sont identiques à ceux des commandes PCF. Lorsque des commandes MQSC sont utilisées pour définir, modifier ou afficher des attributs, d'autres noms abrégés sont utilisés ; voir Commandes MQSC pour plus d'informations.

Attribut	Description
<a href="#">AlterationDate</a>	Date de la dernière modification de la définition
<a href="#">AlterationTime</a>	Heure de la dernière modification de la définition
<a href="#">AppId</a>	Identificateur d'application
<a href="#">AppType</a>	Type d'application
<a href="#">EnvData</a>	Données d'environnement
<a href="#">ProcessDesc</a>	Description du processus
<a href="#">ProcessName</a>	Nom du processus
<a href="#">QSGDisp</a>	Disposition d'un groupe de partage de files d'attente
<a href="#">UserData</a>	Données utilisateur

## AlterationDate (MQCHAR12)

Il s'agit de la date de la dernière modification de la définition. Le format de la date est YYYY-MM-DD, complété par deux blancs de fin pour que la longueur soit de 12 octets.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_ALTERATION\_DATE avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_DATE\_LENGTH.

### ***AlterationTime (MQCHAR8)***

Il s'agit de l'heure à laquelle la définition a été modifiée pour la dernière fois. Le format de l'heure est HH.MM.SS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_ALTERATION\_TIME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_TIME\_LENGTH.

### ***ApplId (MQCHAR256)***

Il s'agit d'une chaîne de caractères qui identifie l'application à démarrer. Ces informations sont destinées à être utilisées par une application de moniteur de déclenchement qui traite les messages de la file d'attente d'initialisation ; elles sont envoyées à la file d'attente d'initialisation dans le cadre du message de déclenchement.

La signification de *ApplId* est déterminée par l'application trigger-monitor. Le moniteur de déclenchement fourni par IBM MQ requiert que *ApplId* soit le nom d'un programme exécutable. Les remarques suivantes s'appliquent aux environnements indiqués:

- Sous z/OS, *ApplId* doit être:
  - Un identificateur de transaction CICS , pour les applications démarrées à l'aide de la transaction CKTI du moniteur de déclenchement CICS
  - Un identificateur de transaction IMS , pour les applications démarrées à l'aide du moniteur de déclenchement IMS CSQQTRMN
- Sous Windows, le nom de programme peut être précédé d'une unité et d'un chemin de répertoire.
- Sous AIX and Linux, le nom de programme peut être précédé d'un chemin de répertoire.

La chaîne de caractères ne peut pas contenir de valeurs nulles. Il est rempli à droite avec des blancs si nécessaire.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_APPL\_ID avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_PROCESS\_APPL\_ID\_LENGTH.

### ***ApplType (MQLONG)***

Identifie la nature du programme à démarrer en réponse à la réception d'un message de déclenchement. Ces informations sont destinées à être utilisées par une application de moniteur de déclenchement qui traite les messages de la file d'attente d'initialisation ; elles sont envoyées à la file d'attente d'initialisation dans le cadre du message de déclenchement.

*ApplType* peut avoir n'importe quelle valeur, mais les valeurs suivantes sont recommandées pour les types standard ; limitez les types d'application définis par l'utilisateur aux valeurs comprises entre MQAT\_USER\_FIRST et MQAT\_USER\_LAST:

#### **MQAT\_AIX**

Application AIX (même valeur que MQAT\_UNIX).

#### **MQAT\_BATCH**

application de traitement par lots

#### **MQAT\_CICS**

Transaction CICS .

#### **MQAT\_IMS**

Application IMS .

**MQAT\_IMS\_BRIDGE**

Application de pont IMS .

**MQAT\_JAVA**

Application Java .

**MQAT\_MVS**

Application MVS ou TSO (même valeur que MQAT\_ZOS).

**MQAT\_OS390**

Application OS/390 (même valeur que MQAT\_ZOS).

**MQAT\_OS400**

Application IBM i .

**MQAT\_UNIX**

Application UNIX .

**MQAT\_INCONNU**

Application de type inconnu.

**Utilisateur\_MQAT**

Application utilisateur.

**MQAT\_WINDOWS**

Application Windows 64 bits.

**MQAT\_WINDOWS\_NT**

Application Windows 32 bits.

**MQAT\_WLM**

Application du gestionnaire de charge de travail z/OS .

**MQAT\_ZOS**

Application z/OS .

**MQAT\_USER\_FIRST**

Valeur la plus faible pour le type d'application défini par l'utilisateur.

**MQAT\_USER\_LAST**

Valeur la plus élevée pour le type d'application défini par l'utilisateur.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_APPL\_TYPE avec l'appel MQINQ.

**EnvData (MQCHAR128)**

Il s'agit d'une chaîne de caractères qui contient des informations relatives à l'environnement concernant l'application à démarrer. Ces informations sont destinées à être utilisées par une application de moniteur de déclenchement qui traite les messages de la file d'attente d'initialisation ; elles sont envoyées à la file d'attente d'initialisation dans le cadre du message de déclenchement.

La signification de *EnvData* est déterminée par l'application trigger-monitor. Le moniteur de déclenchement fourni par IBM MQ ajoute *EnvData* à la liste de paramètres transmise à l'application démarrée. La liste de paramètres comprend la structure MQTMC2 , suivie d'un blanc, suivi de *EnvData* avec les blancs de fin supprimés. Les remarques suivantes s'appliquent aux environnements indiqués:

- Sous z/OS :
  - *EnvData* n'est pas utilisé par les applications de moniteur de déclenchement fournies par IBM MQ.
  - Si ApplType est MQAT\_WLM, vous pouvez fournir des valeurs par défaut dans EnvData pour les zones ServiceName et ServiceStep dans l'en-tête des informations de travail (MQWIH).
- Sous AIX and Linux, *EnvData* peut être défini sur le caractère & pour exécuter l'application démarrée en arrière-plan.

La chaîne de caractères ne peut pas contenir de valeurs nulles. Il est rempli à droite avec des blancs si nécessaire.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_ENV\_DATA avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_PROCESS\_ENV\_DATA\_LENGTH.

### ***ProcessDesc (MQCHAR64)***

Utilisez cette zone pour les commentaires descriptifs. Le contenu de la zone n'est pas significatif pour le gestionnaire de files d'attente, mais ce dernier peut exiger que la zone ne contienne que des caractères pouvant être affichés. Elle ne peut pas contenir de caractères nuls ; si nécessaire, elle est remplie à droite avec des blancs. Dans une installation DBCS, la zone peut contenir des caractères DBCS (avec une longueur de zone maximale de 64 octets).

**Remarque :** Si cette zone contient des caractères qui ne figurent pas dans le jeu de caractères du gestionnaire de files d'attente (tel que défini par l'attribut de gestionnaire de files d'attente **CodedCharSetId**), ces caractères peuvent être convertis de manière incorrecte si cette zone est envoyée à un autre gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_PROCESS\_DESC avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_PROCESS\_DESC\_LENGTH.

### ***ProcessName (MQCHAR48)***

Il s'agit du nom d'une définition de processus définie sur le gestionnaire de files d'attente local.

Chaque définition de processus a un nom différent de celui des autres définitions de processus appartenant au gestionnaire de files d'attente. Toutefois, le nom de la définition de processus peut être identique à celui d'autres objets de gestionnaire de files d'attente de types différents (par exemple, des files d'attente).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_PROCESS\_NAME avec l'appel MQINQ.

La longueur de cet attribut est donnée par MQ\_PROCESS\_NAME\_LENGTH.

### ***QSGDisp (MQLONG)***

Indique la disposition de la définition de processus. La valeur est l'une des suivantes :

#### **MQQSGD\_Q\_DIR**


L'objet possède une disposition de gestionnaire de files d'attente: la définition d'objet est connue uniquement du gestionnaire de files d'attente local ; la définition n'est pas connue des autres gestionnaires de files d'attente du groupe de partage de files d'attente.

Chaque gestionnaire de files d'attente du groupe de partage de files d'attente peut avoir un objet ayant le même nom et le même type que l'objet en cours, mais il s'agit d'objets distincts et il n'y a pas de corrélation entre ces objets. Leurs attributs ne sont pas contraints d'être identiques les uns aux autres.

#### **MQQSGD\_COPY**

L'objet est une copie locale d'une définition d'objet maître qui existe dans le référentiel partagé. Chaque gestionnaire de files d'attente du groupe de partage de files d'attente peut disposer de sa propre copie de l'objet. Au départ, toutes les copies ont les mêmes attributs, mais vous pouvez modifier chaque copie à l'aide des commandes MQSC, de sorte que ses attributs diffèrent de ceux des autres copies. Les attributs des copies sont resynchronisés lorsque la définition principale du référentiel partagé est modifiée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA\_QSG\_DISP avec l'appel MQINQ.

 Cet attribut est pris en charge uniquement sous z/OS.

### ***UserData (MQCHAR128)***

UserData est une chaîne de caractères qui contient des informations utilisateur relatives à l'application à démarrer. Ces informations sont destinées à être utilisées par une application de moniteur de déclenchement qui traite les messages de la file d'attente d'initialisation ou l'application qui est démarrée par le moniteur de déclenchement. Les informations sont envoyées à la file d'attente d'initialisation dans le cadre du message de déclenchement.

La signification de *UserData* est déterminée par l'application trigger-monitor. Le moniteur de déclenchement fourni par IBM MQ transmet *UserData* à l'application démarrée dans le cadre de la liste de paramètres. La liste de paramètres comprend la structure MQTMC2 (contenant *UserData*), suivie d'un blanc, suivie de *EnvData* avec les blancs de fin supprimés.

La chaîne de caractères ne peut pas contenir de valeurs nulles. Il est rempli à droite avec des blancs si nécessaire. Pour Microsoft Windows, la chaîne de caractères ne doit pas contenir de guillemets si la définition de processus doit être transmise à **runmqtrm**.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA\_USER\_DATA avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ\_PROCESS\_USER\_DATA\_LENGTH.

## Codes retour

Pour chaque appel IBM MQ Message Queue Interface (MQI) et IBM MQ Administration Interface (MQAI), un code **d'achèvement** et un code **anomalie** sont renvoyés par le gestionnaire de files d'attente ou par une routine d'exit, pour indiquer la réussite ou l'échec de l'appel.

Les applications ne doivent pas dépendre d'erreurs vérifiées dans un ordre spécifique, sauf lorsqu'elles sont spécifiquement mentionnées. Si plusieurs codes achèvement ou codes raison peuvent provenir d'un appel, l'erreur particulière signalée dépend de l'implémentation.

Les applications qui vérifient la réussite après un appel d'API IBM MQ doivent toujours vérifier le code achèvement. Ne supposez pas la valeur du code achèvement, en fonction de la valeur du code raison.

## Codes achèvement

Le paramètre de code achèvement (*CompCode*) permet à l'appelant de voir rapidement si l'appel a abouti, s'est terminé partiellement ou a échoué. Voici une liste de codes achèvement, avec plus de détails que dans les descriptions d'appel:

### MQCC\_OK

L'appel a abouti ; tous les paramètres de sortie ont été définis. Le paramètre **Reason** a toujours la valeur MQRC\_NONE dans ce cas.

### MQCC\_WARNING

L'appel s'est terminé partiellement. Certains paramètres de sortie peuvent avoir été définis en plus des paramètres de sortie *CompCode* et *Reason* . Le paramètre **Reason** fournit des informations supplémentaires sur l'exécution partielle.

### MQCC\_FAILED

Le traitement de l'appel n'a pas abouti. L'état du gestionnaire de files d'attente reste inchangé, sauf mention contraire. Les paramètres de sortie *CompCode* et *Reason* ont été définis ; les autres paramètres sont inchangés, sauf indication contraire.

Il peut s'agir d'une erreur dans le programme d'application ou d'une situation externe au programme. Par exemple, les droits de l'utilisateur peuvent avoir été révoqués. Le paramètre **Reason** fournit des informations supplémentaires sur l'erreur.

## Codes raison

Le paramètre de code anomalie (*Reason*) qualifie le paramètre de code achèvement (*CompCode*).

S'il n'y a pas de raison particulière à signaler, MQRC\_NONE est renvoyé. Un appel réussi renvoie MQCC\_OK et MQRC\_NONE.

Si le code achèvement est MQCC\_WARNING ou MQCC\_FAILED, le gestionnaire de files d'attente indique toujours une raison de qualification ; des détails sont fournis sous chaque description d'appel.

Lorsque des routines d'exit utilisateur définissent des codes achèvement et des raisons, elles doivent respecter ces règles. En outre, les valeurs de raison spéciale définies par les exits utilisateur doivent être inférieures à zéro, afin de s'assurer qu'elles n'entrent pas en conflit avec les valeurs définies par le gestionnaire de files d'attente. Les exits peuvent définir des raisons déjà définies par le gestionnaire de files d'attente, le cas échéant.

Les codes anomalie apparaissent également dans:

- Zone *Reason* de la structure MQDLH
- Zone *Feedback* de la structure MQMD

Pour obtenir une description complète des codes anomalie, voir [Messages et codes anomalie](#).

## Règles de validation des options MQI

Cette section répertorie les situations qui produisent un code anomalie MQRC\_OPTIONS\_ERROR à partir d'un appel MQOPEN, MQPUT, MQPUT1, MQGET, MQCLOSE ou MQSUB.

### Appel MQOPEN

Pour les options de l'appel MQOPEN:

- Au moins *un* des éléments suivants doit être spécifié:

- MQOO\_BROWSE
- MQOO\_INPUT\_EXCLUSIVE <sup>1</sup>
- MQOO\_INPUT\_SHARED <sup>1</sup>
- MQOO\_INPUT\_AS\_Q\_DEF <sup>1</sup>
- MQOO\_INTERROGATION
- MQOO\_SORTIE
- MQOO\_SET
- MQOO\_BIND\_ON\_OPEN <sup>2</sup>
- MQOO\_BIND\_NOT\_FIXED <sup>2</sup>
- MQOO\_BIND\_ON\_GROUP <sup>2</sup>
- MQOO\_BIND\_AS\_Q\_DEF <sup>2</sup>

- Seul *un* des éléments suivants est autorisé:

- MQOO\_READ\_AHEAD
- MQOO\_NO\_READ\_AHEAD
- MQOO\_READ\_AHEAD\_AS\_Q\_DEF

1. Seul *un* des éléments suivants est autorisé:

- MQOO\_INPUT\_EXCLUSIVE
- MQOO\_INPUT\_SHARED
- MQOO\_INPUT\_AS\_Q\_DEF

2. Seul *un* des éléments suivants est autorisé:

- MQOO\_BIND\_ON\_OPEN
- MQOO\_BIND\_NOT\_FIXED
- MQOO\_BIND\_ON\_GROUPE
- MQOO\_BIND\_AS\_Q\_DEF

**Remarque :** Les options répertoriées précédemment s'excluent mutuellement. Cependant, comme la valeur de MQOO\_BIND\_AS\_Q\_DEF est égale à zéro, le fait de la spécifier avec l'une des deux autres options de liaison n'entraîne pas le code anomalie MQRC\_OPTIONS\_ERROR. MQOO\_BIND\_AS\_Q\_DEF est fourni pour la documentation du programme d'aide.

- Si MQOO\_SAVE\_ALL\_CONTEXT est spécifié, l'une des options MQOO\_INPUT\_ \* doit également être spécifiée.
- Si l'une des options MQOO\_SET\_ \* \_CONTEXT ou MQOO\_PASS\_ \* \_CONTEXT est spécifiée, MQOO\_OUTPUT doit également être spécifié.

- Si MQOO\_CO\_OP est spécifié, MQOO\_BROWSE doit également être spécifié
- Si MQOO\_NO\_MULTICAST est spécifié, MQOO\_OUTPUT doit également être spécifié.

## Appel MQPUT

Pour les options d'insertion de message:

- La combinaison de MQPMO\_SYNCPOINT et MQPMO\_NO\_SYNCPOINT n'est pas autorisée.
- Seul *un* des éléments suivants est autorisé:
  - MQPMO\_CONTEXTE\_PAR\_DÉFAUT
  - MQPMO\_NO\_CONTEXT
  - MQPMO\_PASS\_ALL\_CONTEXT
  - MQPMO\_PASS\_IDENTITY\_CONTEXT
  - MQPMO\_SET\_ALL\_CONTEXT
  - MQPMO\_SET\_IDENTITY\_CONTEXT
- Seul *un* des éléments suivants est autorisé:
  - MQPMO\_ASYNC\_RESPONSE
  - MQPMO\_SYNC\_REPONSE
  - MQPMO\_REPONSE\_AS\_TOPIC\_DEF
  - MQPMO\_REPONSE\_AS\_Q\_DEF
- MQPMO\_ALTERNATE\_USER\_AUTHORITY n'est pas autorisé (il est valide uniquement sur l'appel MQPUT1).

## Appel MQPUT1

Pour les options put-message, les règles sont les mêmes que pour l'appel MQPUT, à l'exception de ce qui suit:

- MQPMO\_ALTERNATE\_USER\_AUTHORITY est autorisé.
- MQPMO\_LOGICAL\_ORDER n'est pas autorisé.

## Appel MQGET

Pour les options get-message:

- Seul *un* des éléments suivants est autorisé:
  - MQGMO\_NO\_SYNCPOINT
  - MQGMO\_SYNCPOINT
  - MQGMO\_SYNCPOINT\_IF\_PERSISTENT
- Seul *un* des éléments suivants est autorisé:
  - MQGMO\_BROWSE\_FIRST
  - MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
  - MQGMO\_BROWSE\_NEXT
  - MQGMO\_MSG\_UNDER\_CURSOR
- MQGMO\_SYNCPOINT n'est pas admise avec l'une des options suivantes:
  - MQGMO\_BROWSE\_FIRST
  - MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
  - MQGMO\_BROWSE\_NEXT
  - MQGMO\_LOCK

- MQGMO\_UNLOCK
- MQGMO\_SYNCPOINT\_IF\_PERSISTENT n'est pas autorisé avec l'un des éléments suivants:
  - MQGMO\_BROWSE\_FIRST
  - MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
  - MQGMO\_BROWSE\_NEXT
  - MQGMO\_COMPLETE\_MSG
  - MQGMO\_UNLOCK
- MQGMO\_MARK\_SKIP\_BACKOUT requiert la spécification de MQGMO\_SYNCPOINT.
- La combinaison de MQGMO\_WAIT et MQGMO\_SET\_SIGNAL n'est pas autorisée.
- Si MQGMO\_LOCK est spécifié, l'un des éléments suivants doit également être spécifié:
  - MQGMO\_BROWSE\_FIRST
  - MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
  - MQGMO\_BROWSE\_NEXT
- Si MQGMO\_UNLOCK est spécifié, seules les valeurs suivantes sont autorisées:
  - MQGMO\_NO\_SYNCPOINT
  - MQGMO\_NO\_WAIT

## Appel MQCLOSE

Pour les options de l'appel MQCLOSE:

- La combinaison de MQCO\_DELETE et MQCO\_DELETE\_PURGE n'est pas autorisée.
- Une seule des options suivantes est autorisée:
  - MQCO\_KEEP\_SUB
  - SOUS-TITRE\_REMOVE\_MQQUE

## Appel MQSUB

Pour les options de l'appel MQSUB:

- Au moins un des éléments suivants doit être spécifié:
  - MQSO\_ALTER
  - MQSO\_RESUME
  - MQSO\_CREER
- Une seule des options suivantes est autorisée:
  - MQSO\_DURABLE
  - MQSO\_NON\_DURABLE

**Remarque :** Les options répertoriées précédemment s'excluent mutuellement. Toutefois, comme la valeur de MQSO\_NON\_DURABLE est égale à zéro, sa spécification avec MQSO\_DURABLE ne génère pas de code anomalie MQRC\_OPTIONS\_ERROR. MQSO\_NON\_DURABLE est fourni pour aider la documentation du programme.

- La combinaison de MQSO\_GROUP\_SUB et de MQSO\_MANAGED n'est pas autorisée.
- MQSO\_GROUP\_SUB requiert la spécification de MQSO\_SET\_CORREL\_ID.
- Une seule des options suivantes est autorisée:
  - ID utilisateur MQSO\_ANY\_USERID
  - ID\_UTILISATEUR\_FIXE\_MQSO\_FIXE
- MQSO\_NEW\_PUBLICATIONS\_ONLY est autorisé en combinaison avec:



- MQSO\_CREER
- MQSO\_ALTER, si MQSO\_NEW\_PUBLICATIONS\_ONLY a été défini sur l'abonnement d'origine
- La combinaison de MQSO\_PUBLICATIONS\_ON\_REQUEST et de SubLevel supérieure à 1 n'est pas autorisée.
- Une seule des options suivantes est autorisée:
  - MQSO\_WILDCARD\_CHAR
  - MQSO\_WILDCARD\_TOPIC
- MQSO\_NO\_MULTICAST requiert la spécification de MQSO\_MANAGED.

## Messages de commande de publication / abonnement en file d'attente

Une application peut utiliser des messages de commande MQRFH2 pour contrôler une application de publication / abonnement en file d'attente.

Une application qui utilise MQRFH2 pour la publication / l'abonnement peut envoyer les messages de commande suivants à SYSTEM.BROKER.CONTROL.QUEUE:

- [«Supprimer le message de publication»](#), à la page 921
- [«Désenregistrer le message d'abonné»](#), à la page 923
- [«Publier le message»](#), à la page 927
- [«Enregistrement du message d'abonnement»](#), à la page 929
- [«Message de demande de mise à jour»](#), à la page 934

Si vous écrivez des applications de publication / abonnement en file d'attente, vous devez comprendre ces messages, le message de réponse du gestionnaire de files d'attente et le descripteur de message (MQMD) ; voir les informations suivantes:

- [«Message de réponse du gestionnaire de files d'attente»](#), à la page 936
- [«Paramètres MQMD pour les publications transmises par un gestionnaire de files d'attente»](#), à la page 942
- [«Paramètres MQMD dans les messages de réponse du gestionnaire de files d'attente»](#), à la page 943
- [«Codes raison de publication / abonnement»](#), à la page 938

Les commandes sont contenues dans un dossier psc dans la zone **NameValueData** de l'en-tête MQRFH2 . Le message qui peut être envoyé par un courtier en réponse à un message de commande est contenu dans un dossier psc1 .

Les descriptions de chaque commande répertorient les propriétés qui peuvent être contenues dans un dossier. Sauf indication contraire, les propriétés sont facultatives et ne peuvent apparaître qu'une seule fois.

Les noms des propriétés sont affichés sous la forme <Command>.

Les valeurs doivent être au format chaîne, par exemple: Publish.

Une constante de type chaîne représentant la valeur d'une propriété est affichée entre parenthèses, par exemple: (MQPSC\_PUBLISH).

Les constantes de chaîne sont définies dans le fichier d'en-tête cmqpsc .h qui est fourni avec le gestionnaire de files d'attente.

## Supprimer le message de publication

Le message de commande **Delete Publication** est envoyé à un gestionnaire de files d'attente à partir d'un diffuseur de publications ou d'un autre gestionnaire de files d'attente pour lui demander de supprimer les publications conservées pour les rubriques spécifiées.

Ce message est envoyé à une file d'attente surveillée par l'interface de publication / abonnement en file d'attente du gestionnaire de files d'attente.

La file d'entrée doit être la file d'attente à laquelle la publication d'origine a été envoyée.

Si vous disposez des droits pour certaines, mais pas pour toutes, des rubriques qui sont spécifiées dans le message de commande **Delete Publication**, seules ces rubriques sont supprimées. Un message **Broker Response** indique les rubriques qui ne sont pas supprimées.

De même, si une commande **Publish** contient plusieurs rubriques, une commande **Delete Publication** correspondant à certaines de ces rubriques, mais pas à toutes, supprime uniquement les publications des rubriques spécifiées dans la commande **Delete Publication**.

Pour plus de détails sur les paramètres de descripteur de message (MQMD) nécessaires lors de l'envoi d'un message de commande au gestionnaire de files d'attente, voir [«Paramètres MQMD pour les publications transmises par un gestionnaire de files d'attente»](#), à la page 942.

## Propriétés

### Commande (MQPSC\_COMMAND)

La valeur est DeletePub (MQPSC\_DELETE\_PUBLICATION).

Cette propriété doit être spécifiée.

### Rubrique > (MQPSC\_TOPIC)

La valeur est une chaîne qui contient une rubrique pour laquelle les publications conservées doivent être supprimées. Des caractères génériques peuvent être inclus dans la chaîne pour supprimer des publications dans plusieurs rubriques.

Cette propriété doit être spécifiée ; elle peut être répétée pour autant de rubriques que nécessaire.

### DelOpt (MQPSC\_DELETE\_OPTION)

La propriété des options de suppression peut prendre l'une des valeurs suivantes:

#### Local (MQPSC\_LOCAL)

Toutes les publications conservées pour les rubriques spécifiées sont supprimées du gestionnaire de files d'attente local (c'est-à-dire du gestionnaire de files d'attente auquel ce message est envoyé), qu'elles aient été publiées avec l'option Local ou non.

Les publications des autres gestionnaires de files d'attente ne sont pas affectées.

#### Aucun (MQPSC\_NONE)

Toutes les options prennent leurs valeurs par défaut. Cela a le même effet que l'omission de la propriété DelOpt. Si d'autres options sont spécifiées en même temps, Aucun est ignoré.

Par défaut, si cette propriété est omise, toutes les publications conservées pour les rubriques spécifiées sont supprimées dans tous les gestionnaires de files d'attente du réseau, qu'elles aient été publiées ou non avec l'option Local.

## Exemple

Voici un exemple de NameValueData pour un message de commande **Delete Publication**. Il est utilisé par l'exemple d'application pour supprimer, au niveau du gestionnaire de files d'attente local, la publication conservée qui contient le score le plus récent dans la correspondance entre Team1 et Team2.

```
<psc>
  <Command>DeletePub</Command>
  <Topic>Sport/Soccer/State/LatestScore/Team1 Team2</Topic>
  <DelOpt>Local</DelOpt>
</psc>
```

## Désenregistrer le message d'abonné

Le message de commande **Deregister Subscriber** est envoyé à un gestionnaire de files d'attente par un abonné, ou par une autre application pour le compte d'un abonné, pour indiquer qu'il ne souhaite plus recevoir de messages correspondant aux paramètres indiqués.

Ce message est envoyé à SYSTEM.BROKER.CONTROL.QUEUE, file d'attente de contrôle du gestionnaire de files d'attente. L'utilisateur doit disposer des droits nécessaires pour placer un message dans cette file d'attente.

Pour plus de détails sur les paramètres de descripteur de message (MQMD) requis lors de l'envoi d'un message de commande au gestionnaire de files d'attente, voir [Paramètres MQMD pour les publications transmises par un gestionnaire de files d'attente](#).

Un abonnement individuel peut être désenregistré en spécifiant les valeurs de rubrique, de point d'abonnement et de filtre correspondantes de l'abonnement d'origine. Si l'une des valeurs n'a pas été spécifiée (c'est-à-dire qu'elle a pris les valeurs par défaut) dans l'abonnement d'origine, elle doit être omise lorsque l'abonnement est désenregistré.

Tous les abonnements d'un abonné ou d'un groupe d'abonnés peuvent être désenregistrés à l'aide de l'option `DeregAll`. Par exemple, si `DeregAll` est spécifié, avec un point d'abonnement (mais pas de rubrique ni de filtre), tous les abonnements de l'abonné sur le point d'abonnement spécifié sont désenregistrés, indépendamment de la rubrique et du filtre. Toute combinaison de rubrique, de filtre et de point d'abonnement est autorisée ; si les trois sont spécifiés, un seul abonnement peut correspondre et l'option `DeregAll` est ignorée.

Le message doit être envoyé par l'abonné qui a enregistré l'abonnement ; ceci est confirmé en vérifiant l'ID utilisateur de l'abonné.

Les abonnements peuvent également être désenregistrés par un administrateur système à l'aide de commandes MQSC ou PCF. Toutefois, les abonnements enregistrés avec une file d'attente dynamique temporaire sont associés à la file d'attente, et pas seulement au nom de la file d'attente. Si la file d'attente est supprimée, explicitement ou par déconnexion de l'application du gestionnaire de files d'attente, il n'est plus possible d'utiliser la commande **Deregister Subscriber** pour désenregistrer les abonnements de cette file d'attente. Les abonnements peuvent être désenregistrés à l'aide du plan de travail du développeur et ils sont supprimés automatiquement par le gestionnaire de files d'attente lors de la prochaine mise en correspondance d'une publication avec l'abonnement ou lors du prochain redémarrage du gestionnaire de files d'attente. Dans des circonstances normales, les applications doivent désenregistrer leurs abonnements avant de supprimer la file d'attente ou de se déconnecter du gestionnaire de files d'attente.

Si un abonné envoie un message pour désenregistrer un abonnement et qu'il reçoit un message de réponse indiquant que le traitement a abouti, certaines publications peuvent encore atteindre la file d'attente de l'abonné si elles étaient traitées par le gestionnaire de files d'attente en même temps que l'abonnement a été désenregistré. Si les messages ne sont pas supprimés de la file d'attente, il peut y avoir une accumulation de messages non traités dans la file d'attente de l'abonné. Si l'application exécute une boucle qui inclut un appel MQGET avec l'ID `CorrelId` approprié après une mise en veille pendant un certain temps, ces messages sont effacés de la file d'attente.

De même, si l'abonné utilise une file d'attente dynamique permanente, désenregistre et ferme la file d'attente avec l'option `MQCO_DELETE_PURGE` sur un appel MQCLOSE, la file d'attente peut ne pas être vide. Si des publications du gestionnaire de files d'attente ne sont pas encore validées lors de la suppression de la file d'attente, un code retour `MQRC_Q_NOT_EMPTY` est émis par l'appel MQCLOSE. L'application peut éviter ce problème en mettant en veille et en émettant à nouveau l'appel MQCLOSE de temps à autre.

### Propriétés

#### Commande (*MQPSC\_COMMAND*)

La valeur est `DeregSub` (*MQPSC\_REGISTER\_SUBSCRIBER*).

Cette propriété doit être spécifiée.

### **Rubrique (MQPSC\_TOPIC)**

La valeur est une chaîne qui contient la rubrique à désenregistrer.

Cette propriété peut, si vous le souhaitez, être répétée si plusieurs rubriques doivent être désenregistrées. Elle peut être omise si DeregAll est spécifié dans <RegOpt>.

Les rubriques spécifiées peuvent être un sous-ensemble de celles qui sont enregistrées si l'abonné souhaite conserver des abonnements pour d'autres rubriques. Les caractères génériques sont autorisés, mais une chaîne de rubrique qui contient des caractères génériques doit correspondre exactement à la chaîne correspondante qui a été spécifiée dans le message de commande

**Deregister Subscriber**.

### **SubPoint (MQPSC\_SUBSCRIPTION\_POINT)**

La valeur est une chaîne qui spécifie le point d'abonnement à partir duquel l'abonnement doit être déconnecté.

Cette propriété ne doit pas être répétée. Elle peut être omise si < Topic> est spécifié ou si DeregAll est spécifié dans <RegOpt>. Si vous omettez cette propriété, les événements suivants se produisent:

- Si vous ne spécifiez **pas** DeregAll, les abonnements correspondant à la propriété < Topic> (et à la propriété < Filter >, le cas échéant) sont désenregistrés du point d'abonnement par défaut.
- Si vous spécifiez DeregAll, tous les abonnements (correspondant aux propriétés < Topic> et < Filter >, le cas échéant) sont désenregistrés de tous les points d'abonnement.

Notez que vous ne pouvez pas spécifier explicitement le point d'abonnement par défaut. Par conséquent, il n'est pas possible de désenregistrer tous les abonnements à partir de ce point d'abonnement uniquement ; vous devez spécifier les rubriques.

### **SubIdentity (MQPSC\_SUBSCRIPTION\_IDENTITY)**

Il s'agit d'une chaîne de longueur variable d'une longueur maximale de 64 caractères. Il est utilisé pour représenter une application ayant un intérêt dans un abonnement. Le gestionnaire de files d'attente gère un ensemble d'identités d'abonné pour chaque abonnement. Chaque abonnement peut permettre à son ensemble d'identités de ne contenir qu'une seule identité ou un nombre illimité d'identités.

Si SubIdentity se trouve dans l'ensemble d'identités de l'abonnement, il est supprimé de l'ensemble. Si l'ensemble d'identités devient vide à la suite de cette opération, l'abonnement est supprimé du gestionnaire de files d'attente, sauf si LeaveOnly est spécifié comme valeur de la propriété RegOpt. Si l'ensemble d'identités contient toujours d'autres identités, l'abonnement n'est pas supprimé du gestionnaire de files d'attente et le flux de publication n'est pas interrompu.

Si SubIdentity est spécifié, mais que SubIdentity ne figure pas dans l'identité définie pour l'abonnement, la commande **Deregister Subscriber** échoue avec le code retour *MQRCCF\_SUB\_IDENTITY\_ERROR*.

### **Filtre (MQPSC\_FILTER)**

La valeur est une chaîne spécifiant le filtre à désenregistrer. Il doit correspondre exactement, y compris la casse et les espaces, à un filtre d'abonnement précédemment enregistré.

Cette propriété peut éventuellement être répétée si plusieurs filtres doivent être désenregistrés. Elle peut être omise si < Topic> est spécifié ou si DeregAll est spécifié dans <RegOpt>.

Les filtres spécifiés peuvent être un sous-ensemble de ceux enregistrés si l'abonné souhaite conserver des abonnements pour d'autres filtres.

### **RegOpt (MQPSC\_REGISTRATION\_OPTION)**

La propriété des options d'enregistrement peut prendre les valeurs suivantes:

#### **DeregAll**

(MQPSC\_DEREGISTER\_ALL)

Tous les abonnements correspondants enregistrés pour cet abonné doivent être désenregistrés.

Si vous spécifiez DeregAll:

- < Topic>, <SubPoint>et < Filtre > peuvent être omis.
- < Topic> et < Filtre > peuvent être répétés, si nécessaire.
- <SubPoint> ne doit pas être répété.

Si vous **ne** spécifiez pas DeregAll:

- < Topic> doit être spécifié et peut être répété si nécessaire.
- <SubPoint> et < Filtre > peuvent être omis.
- <SubPoint> ne doit pas être répété.
- < Filtre > peut être répété, si nécessaire.

Si les rubriques et les filtres sont répétés, tous les abonnements correspondant à toutes les combinaisons des deux sont supprimés. Par exemple, une commande **Deregister Subscriber** qui spécifie trois rubriques et trois filtres tente de supprimer neuf abonnements.

### **IDCorrelAs**

(MQPSC\_CORREL\_ID\_AS\_IDENTITY)

L'ID CorrelId du descripteur de message (MQMD), qui ne doit pas être égal à zéro, est utilisé pour identifier l'abonné. Il doit correspondre à l'ID CorrelId utilisé dans l'abonnement d'origine.

### **FullResp**

(MQPSC\_FULL\_RESPONSE)

Lorsque FullResp est spécifié, tous les attributs de l'abonnement sont renvoyés dans le message de réponse, si la commande n'échoue pas.

Lorsque FullResp est spécifié, DeregAll n'est pas autorisé dans la commande **Deregister Subscriber**. Il n'est pas non plus possible de spécifier plusieurs rubriques. La commande échoue avec le code retour *MQRCCF\_REG\_OPTIONS\_ERROR*, dans les deux cas.

### **LeaveOnly**

(MQPSC\_LEAVE\_ONLY)

Lorsque vous le spécifiez avec une SubIdentity qui se trouve dans l'ensemble d'identités de l'abonnement, SubIdentity est supprimé de l'ensemble d'identités de l'abonnement. L'abonnement n'est pas supprimé du gestionnaire de files d'attente, même si l'ensemble d'identités résultant est vide. Si la valeur SubIdentity ne figure pas dans l'ensemble d'identités, la commande échoue avec le code retour *MQRCCF\_SUB\_IDENTITY\_ERROR*.

Si LeaveOnly est spécifié sans SubIdentity, la commande échoue avec le code retour *MQRCCF\_REG\_OPTIONS\_ERROR*.

Si ni LeaveOnly ni SubIdentity ne sont spécifiés, l'abonnement est supprimé quel que soit le contenu de l'identité définie pour l'abonnement.

### **Aucun**

(MQPSC\_NONE)

Toutes les options prennent leurs valeurs par défaut. Cela a le même effet que l'omission de la propriété des options d'enregistrement. Si d'autres options sont spécifiées en même temps, Aucun est ignoré.

### **VariableUserID**

(MQPSC\_VARIABLE\_USER\_ID)

Lorsqu'elle est spécifiée, l'identité de l'abonné (file d'attente, gestionnaire de files d'attente et ID corrélation) n'est pas limitée à un seul ID utilisateur. Ce comportement diffère du comportement existant du gestionnaire de files d'attente qui associe l'ID utilisateur du message d'enregistrement d'origine à l'identité de l'abonné et empêche ensuite tout autre utilisateur d'utiliser cette identité. Si un nouvel abonné tente d'utiliser la même identité, le code retour *MQRCCF\_DUPLICATE\_SUBSCRIPTION* est renvoyé.

Tout utilisateur peut modifier ou désenregistrer l'abonnement lorsqu'il dispose des droits appropriés, en évitant la vérification existante que l'ID utilisateur doit correspondre à celui de l'abonné d'origine.

Pour ajouter cette option à un abonnement existant, la commande doit provenir du même ID utilisateur que l'abonnement d'origine lui-même.

Si `VariableUserID` est défini pour l'abonnement à désenregistrer, il doit être défini à l'heure de désenregistrement pour indiquer l'abonnement en cours de désenregistrement. Sinon, l'ID utilisateur de la commande **Deregister Subscriber** est utilisé pour identifier l'abonnement. Elle est remplacée, avec les autres identificateurs d'abonné, si un nom d'abonnement est fourni.

Par défaut, si cette propriété est omise, aucune option d'enregistrement n'est définie.

#### **QMgrName (MQPSC\_Q\_MGR\_NAME)**

La valeur est le nom du gestionnaire de files d'attente pour la file d'attente de souscription. Il doit correspondre au `QMgrName` utilisé dans l'abonnement d'origine.

Si cette propriété est omise, la valeur par défaut est le nom `ReplyToQMgr` dans le descripteur de message (MQMD). Si le nom obtenu est vide, le nom du gestionnaire de files d'attente est utilisé par défaut.

#### **QName (MQPSC\_Q\_NAME)**

La valeur est le nom de la file d'attente de souscription. Il doit correspondre au `QName` utilisé dans l'abonnement d'origine.

Si cette propriété est omise, la valeur par défaut est le nom `ReplyToQ` dans le descripteur de message (MQMD), qui ne doit pas être vide.

#### **SubName (MQPSC\_SUBSCRIPTION\_NAME)**

Si vous spécifiez `SubName` dans une commande **Deregister Subscriber**, la valeur `SubName` est prioritaire sur toutes les autres zones d'identificateur à l'exception de l'ID utilisateur, sauf si `VariableUserId` est défini sur l'abonnement lui-même. Si `VariableUserId` n'est pas défini, la commande **Deregister Subscriber** aboutit uniquement si l'ID utilisateur du message de commande correspond à celui de l'abonnement, sinon la commande échoue avec le code retour `MQRCCF_DUPLICATE_IDENTITY`.

S'il existe un abonnement qui correspond à l'identité traditionnelle de cette commande mais qui ne comporte pas de `SubName`, la commande **Deregister Subscriber** échoue avec le code retour `MQRCCF_SUB_NAME_ERROR`. Si une tentative est effectuée pour désenregistrer un abonnement comportant un `SubName` à l'aide d'un message de commande qui correspond à l'identité traditionnelle mais sans `SubName` spécifié, la commande aboutit.

#### **Données SubUser(MQPSC\_SUBSCRIPTION\_USER\_DATA)**

Il s'agit d'une chaîne de texte de longueur variable. La valeur est stockée par le gestionnaire de files d'attente avec l'abonnement mais n'a aucune influence sur la distribution de la publication à l'abonné. La valeur peut être modifiée en s'enregistrant à nouveau dans le même abonnement avec une nouvelle valeur. Cet attribut est destiné à l'utilisation de l'application.

SubUserLes données sont renvoyées dans les informations de métarubrique (`MQCACF_REG_SUB_USER_DATA`) pour un abonnement, si des données `SubUser` sont présentes.

### **Exemple**

Voici un exemple de `NameValueData` pour un message de commande **Deregister Subscriber**. Dans cet exemple, l'exemple d'application désenregistre son abonnement aux rubriques qui contiennent le score le plus récent pour toutes les correspondances. L'identité de l'abonné, y compris `CorrelId`, est extraite des valeurs par défaut du MQMD.

```
<psc>
  <Command>DeregSub</Command>
  <RegOpt>CorrelAsId</RegOpt>
  <Topic>Sport/Soccer/State/LatestScore/#</Topic>
</psc>
```

## Publier le message

Le message de commande **Publish** est inséré dans une file d'attente, ou depuis un gestionnaire de files d'attente vers un abonné, pour publier des informations sur une ou plusieurs rubriques spécifiées.

Le droit de placer un message dans une file d'attente et le droit de publier des informations sur une ou plusieurs rubriques spécifiées sont nécessaires.

Si l'utilisateur est autorisé à publier des informations sur certaines, mais pas sur toutes, rubriques, seules ces rubriques sont utilisées pour la publication ; une réponse d'avertissement indique les rubriques qui ne sont pas utilisées pour la publication.

Si un abonné possède des abonnements correspondants, le gestionnaire de files d'attente transmet le message **Publish** aux files d'attente d'abonné définies dans les messages de commande **Register Subscriber** correspondants.

Voir [Message de réponse du gestionnaire de files d'attente](#) pour plus de détails sur les paramètres de descripteur de message (MQMD) requis lors de l'envoi d'un message de commande au gestionnaire de files d'attente et utilisés lorsqu'un gestionnaire de files d'attente transmet une publication à un abonné.

Le gestionnaire de files d'attente réachemine le message **Publish** vers d'autres gestionnaires de files d'attente du réseau qui possèdent des abonnements correspondants, sauf s'il s'agit d'une publication locale.

Les données de publication, le cas échéant, sont incluses dans le corps du message. Les données peuvent être décrites dans un dossier <mcd> dans la zone NameValueData de l'en-tête MQRFH2 .

## Propriétés

### Commande (*MQPSC\_COMMAND*)

La valeur est Publish (*MQPSC\_PUBLISH*).

Cette propriété doit être spécifiée.

### Rubrique (*MQPSC\_TOPIC*)

La valeur est une chaîne qui contient une rubrique qui catégorise cette publication. Aucun caractère générique n'est autorisé.

Vous devez ajouter la rubrique à la liste de noms SYSTEM.QPUBSUB.QUEUE.NAMELIST, voir [Ajout d'un flux](#) pour savoir comment effectuer cette tâche.

Cette propriété doit être spécifiée et peut éventuellement être répétée pour autant de rubriques que nécessaire.

### SubPoint (*MQPSC\_SUBSCRIPTION\_POINT*)

Point d'abonnement sur lequel la publication est publiée.

Dans WebSphere Event Broker 6.0, la valeur de la propriété <SubPoint> est la valeur de l'attribut de point d'abonnement du noeud Publication qui gère la publication.

Dans IBM WebSphere MQ 7.0.1, la valeur de la propriété <SubPoint> doit correspondre au nom d'un point d'abonnement. Voir [Ajout d'un point d'abonnement](#).

### PubOpt (*MQPSC\_PUBLICATION\_OPTION*)

La propriété des options de publication peut prendre les valeurs suivantes:

#### RetainPub

(*MQPSC\_RETAIN\_PUB*)

Le gestionnaire de files d'attente doit conserver une copie de la publication. Si cette option n'est pas définie, la publication est supprimée dès que le gestionnaire de files d'attente l'a envoyée à tous ses abonnés actuels.

#### IsRetainedconservée

(*MQPSC\_IS\_RETAINED\_PUB*)

(ne peut être défini que par un gestionnaire de files d'attente.) Cette publication a été conservée par le gestionnaire de files d'attente. Le gestionnaire de files d'attente définit cette option pour signaler à un abonné que cette publication a été publiée précédemment et qu'elle a été conservée, à condition que l'abonnement ait été enregistré avec l'option `InformIfConservation`. Elle est définie uniquement en réponse à un message de commande `Register Subscriber` ou `Request Update`. Cette option n'est pas définie pour les publications conservées qui sont envoyées directement aux abonnés.

### **Local**

(*MQPSC\_LOCAL*)

Cette option indique au gestionnaire de files d'attente que cette publication ne doit pas être envoyée à d'autres gestionnaires de files d'attente. Tous les abonnés enregistrés sur ce gestionnaire de files d'attente reçoivent cette publication s'ils possèdent des abonnements correspondants.

### **OtherSubsuniquelement**

(*MQPSC\_OTHER\_SUBS\_ONLY*)

Cette option permet un traitement plus simple des applications de type conférence, où un diffuseur de publications est également abonné à la même rubrique. Il indique au gestionnaire de files d'attente de ne pas envoyer la publication à la file d'attente de l'abonné du diffuseur de publications même s'il possède un abonnement correspondant. La file d'attente d'abonné du diffuseur de publications se compose de son `QMGrName`, de son `QName` et de son `CorrelId` facultatif, comme décrit dans la liste suivante.

### **IDCorrelAs**

(*MQPSC\_CORREL\_ID\_AS\_IDENTITY*)

`CorrelId` dans le `MQMD` (qui doit être différent de zéro) fait partie de la file d'attente d'abonné du diffuseur de publications, dans les applications où le diffuseur de publications est également un abonné.

### **Aucun**

(*MQPSC\_NONE*)

Toutes les options prennent leurs valeurs par défaut. Cette opération a le même effet que l'omission de la propriété des options de publication. Si d'autres options sont spécifiées en même temps, `Aucun` est ignoré.

Vous pouvez avoir plusieurs options de publication en introduisant des éléments `<PubOpt>` supplémentaires.

Par défaut, si cette propriété est omise, aucune option de publication n'est définie.

### **PubTime (MQPSC\_PUBLISH\_TIMESTAMP)**

La valeur est un horodatage de publication facultatif défini par le diffuseur de publications. Sa longueur est de 16 caractères avec le format suivant:

```
YYYYMMDDHHMMSSTH
```

en utilisant le temps universel. Ces informations ne sont pas vérifiées par le gestionnaire de files d'attente avant d'être envoyées aux abonnés.

### **SeqNum (MQPSC\_SEQUENCE\_NUMBER)**

La valeur est un numéro de séquence facultatif défini par le diffuseur de publications.

Il doit être incrémenté de 1 à chaque publication. Toutefois, cela n'est pas vérifié par le gestionnaire de files d'attente, qui se contente de transmettre ces informations aux abonnés.

Si des publications d'une même rubrique sont publiées dans des gestionnaires de files d'attente interconnectés différents, il incombe aux diffuseurs de s'assurer que les numéros de séquence, s'ils sont utilisés, sont significatifs.



### QMgrName (MQPSC\_Q\_MGR\_NAME)

La valeur est une chaîne contenant le nom du gestionnaire de files d'attente pour la file d'attente d'abonné du diffuseur de publications, dans les applications où le diffuseur de publications est également un abonné (voir `OtherSubsuniquement`).

Si cette propriété est omise, la valeur par défaut est le nom `ReplyToQMgr` dans le descripteur de message (MQMD). Si le nom obtenu est vide, le nom du gestionnaire de files d'attente est utilisé par défaut.

### QName (MQPSC\_Q\_NAME)

La valeur est une chaîne contenant le nom de la file d'attente d'abonné du diffuseur de publications, dans les applications où le diffuseur de publications est également abonné (voir `OtherSubsuniquement`).

Si cette propriété est omise, la valeur par défaut est le nom `ReplyToQ` dans le descripteur de message (MQMD), qui ne doit pas être vide si `OtherSubsOnly` est défini.

## Exemple

Voici quelques exemples de *NameValueData* pour un message de commande **Publish**.

Le premier exemple concerne une publication envoyée par le simulateur de correspondance dans l'exemple d'application pour indiquer qu'une correspondance a démarré.

```
<psc>
  <Command>Publish</Command>
  <Topic>Sport/Soccer/Event/MatchStarted</Topic>
</psc>
```

Le deuxième exemple concerne une publication conservée. Le dernier score de la correspondance entre `Team1` et `Team2` est publié.

```
<psc>
  <Command>Publish</Command>
  <PubOpt>RetainPub</PubOpt>
  <Topic>Sport/Soccer/State/LatestScore/Team1 Team2</Topic>
</psc>
```

## Enregistrement du message d'abonnement

Le message de commande **Register Subscriber** est envoyé à un gestionnaire de files d'attente par un abonné, ou par une autre application pour le compte d'un abonné, pour indiquer qu'il souhaite s'abonner à une ou plusieurs rubriques à un point d'abonnement. Un filtre de contenu de message peut également être spécifié.

Dans les expressions de filtre de publication / abonnement, l'imbrication de parenthèses entraîne une diminution exponentielle des performances. Evitez d'imbriquer des parenthèses à une profondeur supérieure à environ 6.

Le message est envoyé à `SYSTEM.BROKER.CONTROL.QUEUE`, qui est la file d'attente de contrôle du gestionnaire de files d'attente. Le droit d'insertion d'un message dans cette file d'attente est requis, en plus du droit d'accès (défini par l'administrateur système du gestionnaire de files d'attente) pour la ou les rubriques de l'abonnement.

Si l'utilisateur dispose de droits sur certaines rubriques, mais pas sur toutes, seules les personnes disposant de droits d'accès sont enregistrées ; une réponse d'avertissement indique celles qui ne sont pas enregistrées.

Pour plus de détails sur les paramètres de descripteur de message (MQMD) nécessaires lors de l'envoi d'un message de commande au gestionnaire de files d'attente, voir [«Paramètres MQMD dans les messages de commande envoyés au gestionnaire de files d'attente»](#), à la page 941.

Si la file d'attente de réponses est une file d'attente dynamique temporaire, l'abonnement est désenregistré automatiquement par le gestionnaire de files d'attente lorsque la file d'attente est fermée.

## Propriétés

### Commande (*MQPSC\_COMMAND*)

La valeur est RegSub (*MQPSC\_REGISTER\_SUBSCRIBER*). Cette propriété doit être spécifiée.

### Rubrique (*MQPSC\_TOPIC*)

Rubrique pour laquelle l'abonné souhaite recevoir des publications. Les caractères génériques peuvent être spécifiés dans le cadre de la rubrique.

Si vous utilisez la commande MQSC **display sub** pour examiner l'abonnement créé de cette manière, la valeur de la balise < Topic> s'affiche en tant que propriété TOPICSTR de l'abonnement.

Cette propriété est obligatoire et peut éventuellement être répétée pour autant de rubriques que nécessaire.

### SubPoint (*MQPSC\_SUBSCRIPTION\_POINT*)

La valeur est le point d'abonnement auquel l'abonnement est associé.

Si cette propriété est omise, le point d'abonnement par défaut est utilisé.

Dans WebSphere Event Broker 6.0, la valeur de la propriété <SubPoint> doit correspondre à la valeur de l'attribut de point d'abonnement des noeuds de publication abonnés.

Dans IBM WebSphere MQ 7.0.1, la valeur de la propriété <SubPoint> doit correspondre au nom d'un point d'abonnement. Voir [Ajout d'un point d'abonnement](#).

### Filtre (*MQPSC\_FILTER*)

La valeur est une expression SQL utilisée comme filtre sur le contenu des messages de publication. Si une publication sur la rubrique spécifiée correspond au filtre, elle est envoyée à l'abonné. Cette propriété correspond à la chaîne de sélection utilisée dans les appels MQSUB et MQOPEN. Pour plus d'informations, voir [Sélection sur le contenu d'un message](#)

Si cette propriété est omise, aucun filtrage de contenu n'est effectué.

### RegOpt (*MQPSC\_REGISTRATION\_OPTION*)

Cette propriété Options d'enregistrement peut prendre les valeurs suivantes:

#### AddName

(*MQPSC\_ADD\_NAME*)

Lorsqu'il est spécifié pour un abonnement existant qui correspond à l'identité traditionnelle de cette commande Register Subscription, mais sans valeur SubName en cours, le SubName spécifié dans cette commande est ajouté à l'abonnement.

Si AddName est spécifié, la zone SubName est obligatoire, sinon MQRCCF\_REG\_OPTIONS\_ERROR est renvoyé.

#### IDCorrelAs

(*MQPSC\_CORREL\_ID\_AS\_IDENTITY*)

CorrelId dans le descripteur de message (MQMD) est utilisé lors de l'envoi de publications correspondantes à la file d'attente de l'abonné. CorrelId ne doit pas être égal à zéro,

#### FullResp

(*MQPSC\_FULL\_RESPONSE*)

Lorsqu'il est spécifié, tous les attributs de l'abonnement sont renvoyés dans le message de réponse, si la commande n'échoue pas.

FullResp est valide uniquement lorsque le message de commande fait référence à un abonnement unique. Par conséquent, une seule rubrique est autorisée dans la commande ; sinon, la commande échoue avec le code retour MQRCCF\_REG\_OPTIONS\_ERROR.

**InformIfRet**

(MQPSC\_INFORM\_IF\_RETAINED)

Le gestionnaire de files d'attente informe l'abonné si une publication est conservée lorsqu'il envoie un message de publication en réponse à un message de commande **Register Subscriber** ou **Request Update**. Pour ce faire, le gestionnaire de files d'attente inclut l'option de publication `IsRetainedPub` dans le message.

**JoinExcl**

(MQPSC\_JOIN\_EXCLUSIVE)

Cette option indique que la `SubIdentity` spécifiée doit être ajoutée en tant que membre exclusif de l'ensemble d'identités pour l'abonnement et qu'aucune autre identité ne peut être ajoutée à l'ensemble.

Si l'identité a déjà rejoint `shared` et est la seule entrée de l'ensemble, l'ensemble est remplacé par un verrou exclusif détenu par cette identité. Sinon, si l'abonnement possède actuellement d'autres identités dans l'ensemble d'identités (avec accès partagé), la commande échoue avec le code retour `MQRCCF_SUBSCRIPTION_IN_USE`.

**JoinShared**

(MQPSC\_JOIN\_SHARED)

Cette option indique que la `SubIdentity` spécifiée doit être ajoutée à l'ensemble d'identités de l'abonnement.

Si l'abonnement est actuellement verrouillé exclusivement (à l'aide de l'option `JoinExcl`), la commande échoue avec le code retour `MQRCCF_SUBSCRIPTION_LOCKED`, sauf si l'identité de l'abonnement verrouillé est identique à celle de ce message de commande. Dans ce cas, le verrou est automatiquement remplacé par un verrou partagé.

**Local**

(MQPSC\_LOCAL)

L'abonnement est local et n'est pas distribué aux autres gestionnaires de files d'attente du réseau. Les publications effectuées dans d'autres gestionnaires de files d'attente ne sont pas distribuées à cet abonné, à moins qu'il ne dispose également d'un abonnement global correspondant.

**NewPubsuniquement**

(MQPSC\_NEW\_PUBS\_ONLY)

Les publications conservées qui existent au moment de l'enregistrement de l'abonnement ne sont pas envoyées à l'abonné ; seules les nouvelles publications sont envoyées.

Si un abonné se réenregistre et modifie cette option de sorte qu'elle ne soit plus définie, une publication qui lui a déjà été envoyée peut être envoyée à nouveau.

**NoAlter**

(MQPSC\_NO\_ALTER)

Les attributs d'un abonnement correspondant existant ne sont pas modifiés.

Lorsqu'un abonnement est créé, cette option est ignorée. Toutes les autres options spécifiées s'appliquent au nouvel abonnement.

Si `SubIdentity` possède également l'une des options de jointure (`JoinExcl` ou `JoinShared`) spécifié, l'identité est ajoutée à l'ensemble d'identités, que `NoAlter` soit spécifié ou non.

**Aucun**

(MQPSC\_NONE)

Toutes les options d'enregistrement prennent leurs valeurs par défaut.

Si l'abonné est déjà enregistré, ses options sont réinitialisées à leurs valeurs par défaut (notez que cela n'a pas le même effet que l'omission de la propriété des options d'enregistrement) et l'expiration de l'abonnement est mise à jour à partir du MQMD du message **Register Subscriber**.

Si d'autres options d'enregistrement sont spécifiées en même temps, Aucun est ignoré.

### **NonPers**

(MQPSC\_NON\_PERSISTENT)

Les publications correspondant à cet abonnement sont distribuées à l'abonné sous forme de messages non persistants.

### **Pers**

(MQPSC\_persISTENT)

Les publications correspondant à cet abonnement sont distribuées à l'abonné sous forme de messages persistants.

### **PersAsPub**

(MQPSC\_PERSISTENT\_AS\_PUBLISH)

Les publications correspondant à cet abonnement sont distribuées à l'abonné avec la persistance spécifiée par le diffuseur de publications. Il s'agit du comportement par défaut.

### **File d'attentePersAs**

(MQPSC\_PERSISTENT\_AS\_Q)

Les publications correspondant à cet abonnement sont distribuées à l'abonné avec la persistance spécifiée dans la file d'attente de l'abonné.

### **PubOnReqOnly**

(MQPSC\_PUB\_ON\_REQUEST\_ONLY)

Le gestionnaire de files d'attente n'envoie pas de publications à l'abonné, sauf en réponse à un message de commande **Request Update**.

### **VariableUserID**

(MQPSC\_VARIABLE\_USER\_ID)

Lorsqu'elle est spécifiée, l'identité de l'abonné (file d'attente, gestionnaire de files d'attente et ID corrélation) n'est pas limitée à un seul ID utilisateur. Ce comportement diffère du comportement existant du gestionnaire de files d'attente qui associe l'ID utilisateur du message d'enregistrement d'origine à l'identité de l'abonné et empêche ensuite tout autre utilisateur d'utiliser cette identité. Si un nouvel abonné tente d'utiliser la même identité, *MQRCCF\_DUPLICATE\_SUBSCRIPTION* est renvoyé.

Cela permet à n'importe quel utilisateur de modifier ou de désenregistrer l'abonnement si l'utilisateur dispose des droits appropriés. Il n'est donc pas nécessaire de vérifier que l'ID utilisateur correspond à celui de l'abonné d'origine.

Pour ajouter cette option à un abonnement existant, la commande doit provenir du même ID utilisateur que l'abonnement d'origine lui-même.

Si `VariableUserId` est défini pour l'abonnement de la commande **Request Update**, il doit être défini lors de la mise à jour de la demande pour indiquer l'abonnement auquel il est fait référence. Sinon, l'ID utilisateur de la commande **Request Update** est utilisé pour identifier l'abonnement. Elle est remplacée, avec les autres identificateurs d'abonné, si un nom d'abonnement est fourni.

Si un message de commande **Register Subscriber** sans ce jeu d'options fait référence à un abonnement existant ayant ce jeu d'options, l'option est supprimée de cet abonnement et l'ID utilisateur de l'abonnement est désormais corrigé. S'il existe déjà un abonné ayant la même identité (file d'attente, gestionnaire de files d'attente et identificateur de corrélation) mais avec un ID utilisateur différent associé, la commande échoue avec le code retour *MQRCCF\_DUPLICATE\_IDENTITY* car un seul ID utilisateur peut être associé à une identité d'abonné.

Si la propriété des options d'enregistrement est omise et que l'abonné est déjà enregistré, ses options d'enregistrement ne sont pas modifiées et l'expiration de l'abonnement est mise à jour à partir du MQMD du message **Register Subscriber**.

Si l'abonné n'est pas déjà enregistré, un nouvel abonnement est créé avec toutes les options d'enregistrement prenant leurs valeurs par défaut.

Les valeurs par défaut sont `PeRsAsPub` et aucune autre option n'est définie.

#### **QMgrName (MQPSC\_Q\_MGR\_NAME)**

La valeur est le nom du gestionnaire de files d'attente pour la file d'attente de souscription, à laquelle les publications correspondantes sont envoyées par le gestionnaire de files d'attente.

Si cette propriété est omise, la valeur par défaut est le nom `ReplyToQMgr` dans le descripteur de message (MQMD). Si le nom obtenu est vide, la valeur par défaut est `QMgrNamedu` du gestionnaire de files d'attente.

#### **QName (MQPSC\_Q\_NAME)**

La valeur est le nom de la file d'attente de souscription, à laquelle les publications correspondantes sont envoyées par le gestionnaire de files d'attente.

Si cette propriété est omise, la valeur par défaut est le nom `ReplyToQ` dans le descripteur de message (MQMD), qui ne doit pas être vide dans ce cas.

Si la file d'attente est une file d'attente dynamique temporaire, distribution non permanente des publications ( `NonPeRs` ) doit être spécifié dans la propriété `<RegOpt>` .

Si la file d'attente est une file d'attente dynamique temporaire, l'abonnement est désenregistré automatiquement par le gestionnaire de files d'attente lorsque la file d'attente est fermée.

#### **SubName (MQPSC\_SUBSCRIPTION\_NAME)**

Il s'agit d'un nom donné à un abonnement particulier. Vous pouvez l'utiliser à la place du gestionnaire de files d'attente, de la file d'attente et du `CorrelId` facultatif pour faire référence à un abonnement.

Si un abonnement existe déjà avec ce **SubName** , tous les autres attributs de l'abonnement (Topic, QMgrName, QName, CorrelId, UserId, RegOpts, UserSubData et Expiry) sont remplacés par les attributs, s'ils sont spécifiés, qui sont transmis dans le nouveau message de commande `Register Subscriber` . Toutefois, si **SubName** est utilisé alors qu'aucune zone QName n'est spécifiée et qu'une file d'attente `ReplyToest` spécifiée dans l'en-tête MQMD, la file d'attente de souscription est remplacée par la file d'attente `ReplyTo`.

Si un abonnement correspondant à l'identité traditionnelle de cette commande existe déjà, mais qu'il n'a pas de **SubName** , la commande d'enregistrement échoue avec le code retour `MQRCCF_DUPLICATE_SUBSCRIPTION`, sauf si l'option **AddName** est spécifiée.

Si vous essayez de modifier un abonnement nommé existant à l'aide d'une autre commande `Register Subscriber` qui spécifie le même **SubName** et les valeurs de Topic, QMgrName, QName et CorrelId dans la nouvelle commande correspondent à un abonnement existant différent, avec ou sans SubName défini, la commande échoue avec le code retour `MQRCCF_DUPLICATE_SUBSCRIPTION`. Cela permet d'éviter que deux noms d'abonnement se réfèrent au même abonnement.

#### **SubIdentity (MQPSC\_SUBSCRIPTION\_IDENTITY)**

Cette chaîne est utilisée pour représenter une application ayant un intérêt dans un abonnement. Il s'agit d'une chaîne de caractères de longueur variable d'une longueur maximale de 64 caractères, facultative. Le gestionnaire de files d'attente gère un ensemble d'identités d'abonné pour chaque abonnement. Chaque abonnement peut autoriser son ensemble d'identités à contenir une seule identité ou un nombre illimité d'identités (voir les options **JoinShared** et **JoinExcl** ).

Une commande d'abonnement qui spécifie l'option **JoinShared** ou **JoinExcl** ajoute l'option **SubIdentity** à l'ensemble d'identités de l'abonnement, si ce n'est pas déjà fait et si l'ensemble d'identités existant autorise une telle action, c'est-à-dire qu'aucun autre abonné ne s'est joint exclusivement ou que l'ensemble d'identités est vide.

Toute modification des attributs de l'abonnement résultant d'une commande `Register Subscriber` dans laquelle une **SubIdentity** est spécifiée aboutit uniquement s'il s'agit du seul membre de l'ensemble des identités de cet abonnement. Sinon, la commande échoue avec le code retour `MQRCCF_SUBSCRIPTION_IN_USE`. Cela empêche les attributs d'un abonnement de changer sans que d'autres abonnés intéressés soient informés.

Si vous spécifiez une chaîne de caractères de plus de 64 caractères, la commande échoue avec le code retour *MQRCCF\_SUB\_IDENTITY\_ERROR*.

### **Données SubUser(MQPSC\_SUBSCRIPTION\_USER\_DATA)**

Il s'agit d'une chaîne de texte de longueur variable. La valeur est stockée par le gestionnaire de files d'attente avec l'abonnement, mais n'a aucune influence sur la distribution des publications à l'abonné. La valeur peut être modifiée en s'enregistrant à nouveau dans le même abonnement avec une nouvelle valeur. Cet attribut est destiné à l'utilisation de l'application.

**SubUserData** est renvoyé dans les informations de métarubrique (*MQCACF\_REG\_SUB\_USER\_DATA*) pour un abonnement, le cas échéant.

Si vous spécifiez plusieurs valeurs d'option d'enregistrement *NonPers*, *PersAsPub*, *PersAsQueue*, and *Pers*, seule la dernière est utilisée. Vous ne pouvez pas combiner ces options dans un abonnement individuel.

## **Exemple**

Voici un exemple de `NameValueData` pour un message de commande **Register Subscriber**. Dans l'exemple d'application, le service de résultats utilise ce message pour enregistrer un abonnement aux rubriques contenant les derniers scores dans toutes les correspondances, avec l'option *Persistent as publish* définie. L'identité de l'abonné, y compris `CorrelId`, est extraite des valeurs par défaut du MQMD.

```
<psc>
  <Command>RegSub</Command>
  <RegOpt>PersAsPub</RegOpt>
  <RegOpt>CorrelAsId</RegOpt>
  <Topic>Sport/Soccer/State/LatestScore/#</Topic>
</psc>
```

## **Message de demande de mise à jour**

Le message de commande **Request Update** est envoyé d'un abonné à un gestionnaire de files d'attente pour demander les publications conservées en cours pour la rubrique et le point d'abonnement spécifiés qui correspondent au filtre donné (facultatif).

Ce message est envoyé à *SYSTEM.BROKER.CONTROL.QUEUE*, file d'attente de contrôle du gestionnaire de files d'attente. Le droit d'insertion d'un message dans cette file d'attente est requis, en plus du droit d'accès à la rubrique dans la mise à jour de la demande ; ce droit est défini par l'administrateur système du gestionnaire de files d'attente.

Cette commande est normalement utilisée si l'abonné a spécifié l'option *PubOnReqOnly* lors de son enregistrement. Si le gestionnaire de files d'attente possède des publications conservées correspondantes, elles sont envoyées à l'abonné. Si le gestionnaire de files d'attente ne possède pas de publications conservées correspondantes, la demande échoue avec le code retour *MQRCCF\_NO\_RETAINED\_MSG*. Le demandeur doit avoir préalablement enregistré un abonnement avec les mêmes valeurs de rubrique, `SubPoint` et de filtre.

## **Propriétés**

### **Commande (MQPSC\_COMMAND)**

La valeur est `ReqUpdate` (*MQPSC\_REQUEST\_UPDATE*). Cette propriété doit être spécifiée.

### **Rubrique (MQPSC\_TOPIC)**

La valeur est la rubrique que l'abonné demande ; les caractères génériques sont autorisés.

Cette propriété doit être spécifiée, mais une seule occurrence est autorisée dans ce message.

### **SubPoint (MQPSC\_SUBSCRIPTION\_POINT)**

La valeur est le point d'abonnement auquel l'abonnement est associé.

Si cette propriété est omise, le point d'abonnement par défaut est utilisé.

### **Filtre (MQPSC\_FILTER)**

La valeur est une expression ESQL utilisée comme filtre sur le contenu des messages de publication. Si une publication sur la rubrique spécifiée correspond au filtre, elle est envoyée à l'abonné.

La propriété < Filtre > doit avoir la même valeur que celle spécifiée sur l'abonnement d'origine pour lequel vous demandez une mise à jour.

Si cette propriété est omise, aucun filtrage de contenu n'est effectué.

### **RegOpt (MQPSC\_REGISTRATION\_OPTION)**

La propriété des options d'enregistrement peut prendre la valeur suivante:

#### **IDCorrelAs**

(MQPSC\_CORREL\_ID\_AS\_IDENTITY)

Le CorrelId du descripteur de message (MQMD), qui doit être différent de zéro, est utilisé lors de l'envoi de publications correspondantes à la file d'attente de l'abonné.

#### **Aucun**

(MQPSC\_NONE)

Toutes les options prennent leurs valeurs par défaut. Cela a le même effet que l'omission de la propriété <RegOpt> . Si d'autres options sont spécifiées en même temps, Aucun est ignoré.

#### **VariableUserID**

(MQPSC\_VARIABLE\_USER\_ID)

Lorsqu'elle est spécifiée, l'identité de l'abonné (file d'attente, gestionnaire de files d'attente et ID corrélation) n'est pas limitée à un seul ID utilisateur. Ce comportement diffère du comportement existant du gestionnaire de files d'attente qui associe l'ID utilisateur du message d'enregistrement d'origine à l'identité de l'abonné et empêche ensuite tout autre utilisateur d'utiliser cette identité. Si un nouvel abonné tente d'utiliser la même identité, la commande échoue avec le code retour *MQRCCF\_DUPLICATE\_SUBSCRIPTION*.

Cela permet à tout utilisateur de modifier ou de désenregistrer l'abonnement lorsqu'il dispose des droits appropriés. Par conséquent, il n'est pas nécessaire de vérifier que l'ID utilisateur correspond à celui de l'abonné d'origine.

Pour ajouter cette option à un abonnement existant, la commande doit provenir du même ID utilisateur que l'abonnement d'origine.

Si VariableUserId est défini pour l'abonnement de la commande **Request Update** , il doit être défini lors de la mise à jour de la demande pour indiquer l'abonnement auquel il est fait référence. Sinon, l'ID utilisateur de la commande **Request Update** est utilisé pour identifier l'abonnement. Elle est remplacée, avec les autres identificateurs d'abonné, si un nom d'abonnement est fourni.

Par défaut, si cette propriété est omise, aucune option d'enregistrement n'est définie.

### **QMgrName (MQPSC\_Q\_MGR\_NAME)**

La valeur est le nom du gestionnaire de files d'attente de la file d'attente de souscription, à laquelle la publication conservée correspondante est envoyée par le gestionnaire de files d'attente.

Si cette propriété est omise, la valeur par défaut est le nom ReplyToQMgr dans le descripteur de message (MQMD). Si le nom obtenu est vide, la valeur par défaut est QMgrNamedu gestionnaire de files d'attente.

### **QName (MQPSC\_Q\_NAME)**

La valeur est le nom de la file d'attente de souscription, à laquelle la publication conservée correspondante est envoyée par le gestionnaire de files d'attente.

Si cette propriété est omise, la valeur par défaut est le nom ReplyToQ dans le descripteur de message (MQMD), qui ne doit pas être vide dans ce cas.

### **SubName (MQPSC\_SUBSCRIPTION\_NAME)**

Il s'agit d'un nom donné à un abonnement particulier. Si elle est spécifiée dans une commande **Request Update** , la valeur SubName est prioritaire sur toutes les autres zones d'identificateur

à l'exception de l'ID utilisateur, sauf si `VariableUserId` est défini sur l'abonnement lui-même. Si `VariableUserId` n'est pas défini, la commande *Request Update* aboutit uniquement si l'ID utilisateur du message de commande correspond à celui de l'abonnement. Si l'ID utilisateur du message de commande ne correspond pas à celui de l'abonnement, la commande échoue avec le code retour `MQRCCF_DUPLICATE_IDENTITY`.

Si `VariableUserId` est défini et que l'ID utilisateur est différent de celui de l'abonnement, la commande aboutit si l'ID utilisateur du nouveau message de commande est autorisé à parcourir la file d'attente de flux et à l'insérer dans la file d'attente d'abonné de l'abonnement. Sinon, la commande échoue avec le code retour `MQRCCF_NOT_AUTHORIZED`.

S'il existe un abonnement correspondant à l'identité traditionnelle de cette commande, mais sans `SubName`, la commande **Request Update** échoue avec le code retour `MQRCCF_SUB_NAME_ERROR`.

Si une tentative de demande de mise à jour est effectuée pour un abonnement dont le `SubName` utilise un message de commande qui correspond à l'identité traditionnelle, mais sans `SubName` spécifié, la commande aboutit.

## Exemple

Voici un exemple de `NameValueData` pour un message de commande **Request Update**. Dans l'exemple d'application, le service de résultats utilise ce message pour demander des publications conservées contenant les derniers scores pour toutes les équipes. L'identité de l'abonné, y compris `CorrelId`, est extraite des valeurs par défaut du MQMD.

```
<psc>
  <Command>ReqUpdate</Command>
  <RegOpt>CorrelAsId</RegOpt>
  <Topic>Sport/Soccer/State/LatestScore/#</Topic>
</psc>
```

## Message de réponse du gestionnaire de files d'attente

Un message **Queue Manager Response** est envoyé d'un gestionnaire de files d'attente à la file d'attente `ReplyToQ` d'un diffuseur de publications ou d'un abonné, pour indiquer la réussite ou l'échec d'un message de commande reçu par le gestionnaire de files d'attente si le descripteur de message de commande indique qu'une réponse est requise.

Le message de réponse est contenu dans la zone `NameValueData` de l'en-tête `MQRFH2`, dans un dossier `<pscr>`.

En cas d'avertissement ou d'erreur, le message de réponse contient le dossier `<psc>` du message de commande ainsi que le dossier `<pscr>`. Les données de message, le cas échéant, ne sont pas contenues dans le message de réponse du gestionnaire de files d'attente. Dans le cas d'une erreur, aucun des messages à l'origine d'une erreur n'a été traité ; dans le cas d'un avertissement, certains messages peuvent avoir été traités avec succès.

En cas d'échec lors de l'envoi d'une réponse:

- Pour les messages de publication, le gestionnaire de files d'attente tente d'envoyer la réponse à la file d'attente de rebut IBM MQ en cas d'échec de `MQPUT`. Cela permet d'envoyer la publication aux abonnés même si la réponse ne peut pas être renvoyée au diffuseur de publications.
- Pour les autres messages ou si la réponse de publication ne peut pas être envoyée à la file d'attente de rebut, une erreur est consignée et le message de commande est normalement annulé. Cela dépend de la manière dont le noeud `MQInput` a été configuré.

## Propriétés

### Achèvement (`MQPSCR_COMPLETION`)

Le code achèvement, qui peut prendre l'une des trois valeurs suivantes:

**OK**

La commande a abouti



## AVERTISSEMENT

Commande terminée mais avec avertissement

## ERREUR

La commande a échoué

### Réponse (*MQPSCR\_RESPONSE*)

La réponse à un message de commande, si cette commande a généré le code achèvement `warning` ou `error`. Il contient une propriété `< Reason>` et peut contenir d'autres propriétés qui indiquent la cause de l'avertissement ou de l'erreur.

Dans le cas d'une ou de plusieurs erreurs, il n'y a qu'un seul dossier de réponse, indiquant la cause de la première erreur uniquement. Dans le cas d'un ou de plusieurs avertissements, il existe un dossier de réponses pour chaque avertissement.

### Motif (*MQPSCR\_REASON*)

Code raison qualifiant le code achèvement, si le code achèvement est un avertissement ou une erreur. Il est défini sur l'un des codes d'erreur répertoriés dans l'exemple suivant. La propriété `< Reason>` est contenue dans un dossier `< Response>`. Le code anomalie peut être suivi de toute propriété valide du dossier `<psc>` (par exemple, un nom de rubrique), indiquant la cause de l'erreur ou de l'avertissement. Si vous obtenez un code raison de `????`, vérifiez que les données sont correctes, par exemple, les crochets correspondants (`< >`).

## Exemples

Voici quelques exemples de `NameValueData` dans un message **Queue Manager Response**. Une réponse réussie peut être la suivante:

```
<psc>
  <Completion>ok</Completion>
</psc>
```

Voici un exemple de réponse d'échec ; l'échec est une erreur de filtre. La première chaîne `NameValueData` contient la réponse ; la seconde contient la commande d'origine.

```
<psc>
  <Completion>error</Completion>
  <Response>
    <Reason>3150</Reason>
  </Reponse>
</psc>

<psc>
  ...
  command message (to which
  the queue manager is responding)
  ...
</psc>
```

Voici un exemple de réponse d'avertissement (due à des sujets non autorisés). La première chaîne `NameValueData` contient la réponse ; la deuxième chaîne `NameValueData` contient la commande d'origine.

```
<psc>
  <Completion>warning</Completion>
  <Response>
    <Reason>3081</Reason>
    <Topic>topic1</Topic>
  </Reponse>
  <Response>
    <Reason>3081</Reason>
    <Topic>topic2</Topic>
  </Reponse>
</psc>

<psc>
  ...
```

```

command message (to which
the queue manager is responding)
...
</psc>

```

## Codes raison de publication / abonnement

Ces codes anomalie peuvent être renvoyés dans la zone Raison d'un dossier de réponse de publication / abonnement <pscr> . Les constantes qui peuvent être utilisées pour représenter ces codes dans les langages de programmation C ou C ++ sont également répertoriées.

Les constantes MQRC\_ requièrent le fichier d'en-tête IBM MQ cmqc . h . Les constantes MQRCCF\_ requièrent le fichier d'en-tête IBM MQ cmqfc . h (à l'exception de MQRCCF\_FILTER\_ERROR et MQRCCF\_ILLIC\_USER, qui requièrent le fichier d'en-tête cmqpsc . h ).

Code anomalie et texte	Explication	Emis par
2336 MQRC_RFH_ERREUR DE COMMANDE	Les valeurs valides pour la zone < Command> d'un dossier <psc> sont: RegSub, DeregSub, Publish, DeletePubet ReqUpdate. Toute autre valeur entraîne l'émission de ce code d'erreur.	Toute commande
2337 MQRC_RFH_ERREUR DE PARM	Les dossiers <psc> et <mcd> ont tous deux un ensemble de paramètres valides qui peuvent être spécifiés dans ces deux dossiers. Vérifiez les descriptions de ces dossiers et assurez-vous que vous n'avez pas indiqué de paramètres incorrects.	Toute commande
2338 MQRC_RFH_DUPLICATE_PARM	Certains paramètres (par exemple, Rubrique) d'un dossier <psc> peuvent être répétés, tandis que d'autres (par exemple, Commande) ne peuvent pas être répétés. Vérifiez que vous n'avez pas dupliqué de paramètre non reproductible.	Toute commande
2339 MQRC_RFH_PARM_MISSING	Certains paramètres dans les dossiers <psc> ou <mcd> sont facultatifs et peuvent être omis ; d'autres sont obligatoires et ne doivent pas être omis. Vérifiez que vous avez inclus tous les paramètres obligatoires dans vos dossiers <psc> et <mcd> .	Toute commande
2551 MQRC_SELECTION_NOT_AVAILABLE	Aucun fournisseur de sélection de messages étendue n'était disponible pour déterminer quels abonnés avec un filtre spécifié devaient recevoir la publication.	Publier, enregistrer un abonné et demander la mise à jour
	Aucun fournisseur de sélection de message étendu n'est disponible pour traiter le filtre de l'abonné spécifié.	Enregistrer l'abonné et demander la mise à jour
2554 MQRC_CONTENT_ERREUR	Un fournisseur de sélection de messages étendue a détecté une erreur dans la publication en cours ou conservée.	Publier et demander la mise à jour

Code anomalie et texte	Explication	Emis par
3008 Echec de la commande MQRCCF_COMMAND_FAILED	Une erreur interne s'est produite et a empêché l'exécution correcte de la commande. L'erreur peut se produire si la commande est réémise. Le journal des événements système du gestionnaire de files d'attente contient des informations qui doivent être utilisées pour signaler le problème à IBM.	Toute commande
3072 MQRCCF_TOPIC_ERREUR	Une ou plusieurs des valeurs que vous avez fournies pour le paramètre Topic sont incorrectes. Vérifiez que vos valeurs de rubrique sont conformes aux restrictions spécifiées.	Toute commande
3073 MQRCCF_NOT_REGISTERED	La combinaison de SubPoint, Topic et Filter que vous avez spécifiée dans votre commande DeregSub ou ReqUpdate n'était pas une combinaison avec laquelle vous vous étiez précédemment enregistré ou, pour la commande DeregSub si l'option DeregAll était spécifiée, l'une des propriétés SubPoint, Topic ou Filter n'était pas utilisée pour désenregistrer un abonnement.	Instructions de désenregistrement d'abonné et de mise à jour de demande
3074 MQRCCF_Q_MGR_NAME_ERROR	Le gestionnaire de files d'attente spécifié n'est pas valide, ou le gestionnaire de files d'attente n'est pas disponible ou n'existe pas.	Instructions de désenregistrement d'un abonné, de publication, d'enregistrement d'un abonné et de demande de mise à jour
3076 MQRCCF_Q_NOM_ERREUR	Le nom de file d'attente spécifié n'est pas valide ou la file d'attente n'existe pas sur le gestionnaire de files d'attente spécifié.	Instructions de désenregistrement d'un abonné, de publication, d'enregistrement d'un abonné et de demande de mise à jour
3077 MQRCCF_NO_RETAINED_MSG	Il n'y a pas eu de messages conservés pour la rubrique que vous avez spécifiée. Il peut s'agir ou non d'une erreur, selon la conception de votre programme d'application.	Commande de demande de mise à jour

Code anomalie et texte	Explication	Emis par
3079 MQRCCF_INCORRECT_Q	Les commandes RegSub, DeregSubet ReqUpdate sont toujours envoyées à SYSTEM.BROKER.CONTROL.QUEUE du gestionnaire de files d'attente auquel ils sont destinés. Les commandes de publication et de suppression de publication sont envoyées à la file d'attente d'entrée du flux de messages de publication / abonnement auquel elles sont destinées ; elles sont déterminées lorsque le flux de messages est conçu. Ce code d'erreur est renvoyé si une commande est envoyée à la mauvaise file d'attente.	Toute commande
3080 MQRCCF_CORREL_ID_ERREUR	Vous avez spécifié l'ID CorrelAscomme l'un de vos paramètres RegOpt . Toutefois, la zone CorrelId du MQMD ne contient pas d'identificateur de corrélation valide (c'est-à-dire qu'elle est définie sur MQCI_NONE).	Instructions de désenregistrement d'un abonné et d'enregistrement d'un abonné
3081 MQRCCF_NON_AUTORISÉ	Vous n'êtes pas autorisé à effectuer l'action demandée. Les paramètres d'autorisation du gestionnaire de files d'attente sont gérés par l'administrateur système à l'aide de l'éditeur de hiérarchie des rubriques.	Commandes de publication et d'enregistrement d'abonné
3083 MQRCCF_REG_OPTIONS_ERREUR	Vous avez spécifié un paramètre RegOpt non reconnu dans le dossier <psc> qui contient votre commande RegSub ou DeregSub .	Instructions de désenregistrement d'un abonné et d'enregistrement d'un abonné
3084 MQRCCF_PUB_OPTIONS_ERROR	Vous avez spécifié un paramètre PubOpt non reconnu dans le dossier <psc> qui contient votre commande de publication.	Commande de publication
3087 MQRCCF_DEL_OPTIONS_ERROR	Vous avez spécifié un paramètre DelOpt non reconnu dans le dossier <psc> qui contient la commande DeletePub .	Instruction de suppression d'une publication
3150 MQRCCF_FILTER_ERREUR	La valeur indiquée pour le paramètre Filter n'est pas valide. Consultez la section qui décrit la syntaxe valide pour les expressions de filtre et assurez-vous que votre expression est conforme.	Instructions de désenregistrement d'un abonné, d'enregistrement d'un abonné et de demande de mise à jour

Code anomalie et texte	Explication	Emis par
3151 MQRCCF_UTILISATEUR_FAUT_ERREUR	Un abonnement correspondant à celui indiqué existe déjà ; cependant, il a été enregistré par un autre utilisateur. Un abonnement ne peut être modifié ou désenregistré que par l'utilisateur qui l'a enregistré à l'origine.	Instructions de désenregistrement d'un abonné, d'enregistrement d'un abonné et de demande de mise à jour
3152 MQRCCF_ABONNEMENT_DOUBLON	Un abonnement correspondant existe déjà avec un nom d'abonnement différent.	
3153 MQRCCF_SUB_NAME_ERREUR	Le format du nom d'abonnement n'est pas valide ou un abonnement correspondant existe déjà sans nom d'abonnement.	
3154 MQRCCF_SUB_IDENTITY_ERROR	Le paramètre d'identité d'abonnement est erroné. Soit la valeur fournie dépasse la longueur maximale autorisée, soit l'identité de l'abonnement n'est pas actuellement membre de l'ensemble d'identités de l'abonnement et une option d'enregistrement de jointure n'a pas été spécifiée.	
3155 MQRCCF_SUBSCRIPTION_IN_USE	Une tentative de modification ou de désenregistrement d'un abonnement a été effectuée par un membre de l'ensemble d'identités alors qu'il n'était pas le seul membre de cet ensemble.	
3156 MQRCCF_SUBSCRIPTION_LOCKED	L'abonnement est actuellement verrouillé exclusivement par une autre identité.	
3157 MQRCCF_ALREADY_JOINT	Une option d'enregistrement de jointure a été spécifiée, mais l'identité de l'abonné était déjà membre de l'ensemble d'identités de l'abonnement.	

## Paramètres MQMD dans les messages de commande envoyés au gestionnaire de files d'attente

Les applications qui envoient des messages de commande au gestionnaire de files d'attente utilisent les paramètres suivants des zones du descripteur de message (MQMD). Les zones qui restent comme valeur par défaut ou qui peuvent être définies sur n'importe quelle valeur valide de la manière habituelle ne sont pas répertoriées ici.

### Rapport

Voir `MsgType` et `CorrelId`.

### MsgType

`MsgType` doit être défini sur `MQMT_REQUEST` ou `MQMT_DATAGRAM`. `MQRCCF_MSG_TYPE_ERROR` est renvoyé si `MsgType` n'est pas défini sur l'une de ces valeurs.

MsgType doit être défini sur *MQMT\_REQUEST* pour un message de commande si une réponse est toujours requise. Les indicateurs MQRO\_PAN et MQRO\_NAN de la zone *Rapport* ne sont pas significatifs dans ce cas.

Si MsgType est défini sur *MQMT\_DATAGRAM*, les réponses dépendent de la définition des indicateurs MQRO\_PAN et MQRO\_NAN dans la zone *Rapport* :

- MQRO\_PAN seul signifie que le gestionnaire de files d'attente envoie une réponse uniquement si la commande aboutit.
- MQRO\_NAN seul signifie que le gestionnaire de files d'attente envoie une réponse uniquement si la commande échoue.
- Si une commande se termine avec un avertissement, une réponse est envoyée si MQRO\_PAN ou MQRO\_NAN est défini.
- MQRO\_PAN + MQRO\_NAN signifie que le gestionnaire de files d'attente envoie une réponse indiquant si la commande aboutit ou échoue. Cela a le même effet du point de vue du gestionnaire de files d'attente que de définir MsgType sur *MQMT\_REQUEST*.
- Si ni MQRO\_PAN ni MQRO\_NAN n'est défini, aucune réponse n'est jamais envoyée.

**Format**

Défini sur MQFMT\_RF\_HEADER\_2

**MsgId**

Cette zone est normalement définie sur MQMI\_NONE, de sorte que le gestionnaire de files d'attente génère une valeur unique.

**CorrelId**

Cette zone peut être définie sur n'importe quelle valeur. Si l'identité de l'expéditeur inclut un CorrelId, spécifiez cette valeur, avec MQRO\_PASS\_CORREL\_ID dans la zone *Rapport*, pour vous assurer qu'elle est définie dans tous les messages de réponse envoyés par le gestionnaire de files d'attente à l'expéditeur.

**ReplyToQ**

Cette zone définit la file d'attente à laquelle les réponses, le cas échéant, doivent être envoyées. Il peut s'agir de la file d'attente de l'expéditeur, ce qui présente l'avantage que le paramètre QName peut être omis du message. Toutefois, si des réponses doivent être envoyées à une autre file d'attente, le paramètre QName est nécessaire.

**ReplyToQMgr**

Cette zone définit le gestionnaire de files d'attente pour les réponses. Si vous laissez cette zone vide (valeur par défaut), le gestionnaire de files d'attente local place son propre nom dans cette zone.

## Paramètres MQMD pour les publications transmises par un gestionnaire de files d'attente

Un gestionnaire de files d'attente utilise ces paramètres de zones dans le descripteur de message (MQMD) lorsqu'il envoie une publication à un abonné. Toutes les autres zones du MQMD sont définies sur leurs valeurs par défaut.

**Rapport**

Rapport est défini sur MQRO\_NONE.

**MsgType**

MsgType est défini sur MQMT\_DATAGRAM.

**Expiration**

Expiration est définie sur la valeur du message *Publier* reçu du diffuseur de publications. Dans le cas d'un message conservé, le temps restant est réduit de l'heure approximative à laquelle le message a été envoyé au gestionnaire de files d'attente.

**Format**

Format est défini sur MQFMT\_RF\_HEADER\_2

**MsgId**

MsgId est défini sur une valeur unique.

**CorrelId**

Si CorrelId fait partie de l'identité de l'abonné, il s'agit de la valeur spécifiée par l'abonné lors de l'enregistrement. Sinon, il s'agit d'une valeur différente de zéro choisie par le gestionnaire de files d'attente.

**Priority**

Priority prend la valeur définie par le diffuseur de publications ou comme résolue si le diffuseur de publications a spécifié MQPRI\_PRIORITY\_AS\_Q\_DEF.

**Persistence**

Persistence prend la valeur définie par le diffuseur de publications ou comme résolue si le diffuseur de publications a spécifié MQPER\_PERSISTENCE\_AS\_Q\_DEF, sauf indication contraire dans le message Register Subscriber pour l'abonné auquel cette publication est envoyée.

**ReplyToQ**

ReplyToQ est mis à blanc.

**ReplyToQMgr**

ReplyToGestionnaire de files d'attente est défini sur le nom du gestionnaire de files d'attente.

**UserIdentifieur**

UserIdentifieur est l'identificateur utilisateur de l'abonné, tel que défini lors de l'enregistrement de l'abonné.

**AccountingToken**

AccountingToken est le jeton de comptabilité de l'abonné, tel que défini lors de la première inscription de l'abonné.

**ApplIdentityData**

Les données ApplIdentity sont les données d'identité de l'application de l'abonné, telles qu'elles sont définies lorsque l'abonné est enregistré pour la première fois.

**PutApplType**

PutApplType est défini sur MQAT\_BROKER.

**PutApplName**

PutApplNom est défini sur les 28 premiers caractères du nom du gestionnaire de files d'attente.

**PutDate**

PutDate est la date à laquelle le message a été inséré.

**PutTime**

PutTime est l'heure à laquelle le message a été inséré.

**ApplOriginData**

ApplOriginLes données sont mises à blanc.

## Paramètres MQMD dans les messages de réponse du gestionnaire de files d'attente

Un gestionnaire de files d'attente utilise ces paramètres des zones du descripteur de message (MQMD) lors de l'envoi d'une réponse à un message de publication. Toutes les autres zones du MQMD sont définies sur leurs valeurs par défaut.

**Rapport**

Rapport est défini sur des zéros.

**MsgType**

MsgType est défini sur MQMT\_REPLY.

**Format**

Format est défini sur MQFMT\_RF\_HEADER\_2

**MsgId**

La valeur de `MsgId` dépend des options `Report` du message de commande d'origine. Par défaut, il est défini sur `MQMI_NONE`, de sorte que le gestionnaire de files d'attente génère une valeur unique.

**CorrelId**

La valeur de `CorrelId` dépend des options `Report` du message de commande d'origine. Par défaut, cela signifie que l'ID `CorrelId` est défini sur la même valeur que l'ID `MsgId` du message de commande. Cela peut être utilisé pour corréler les commandes avec leurs réponses.

**Priority**

`Priority` est défini sur la même valeur que dans le message de commande d'origine.

**Persistence**

`Persistence` est défini sur la valeur définie dans le message de commande d'origine.

**Expiration**

`Expiration` est définie sur la même valeur que dans le message de commande d'origine reçu par le gestionnaire de files d'attente.

**PutApplType**

`PutApplType` est défini sur `MQAT_BROKER`.

**PutApplName**

`PutApplNom` est défini sur les 28 premiers caractères du nom du gestionnaire de files d'attente.

D'autres zones de contexte sont définies comme si elles étaient générées avec `MQPMO_PASS_IDENTITY_CONTEXT`.

## Codages de machine

Cette section décrit la structure de la zone *Encoding* dans le descripteur de message.

Pour un récapitulatif des zones de la structure, voir [«MQMD-Descripteur de message»](#), à la page 438 .

La zone *Encoding* est un entier 32 bits qui est divisé en quatre sous-zones distinctes ; ces sous-zones identifient:

- Codage utilisé pour les entiers binaires
- Codage utilisé pour les entiers décimaux condensés
- Codage utilisé pour les nombres à virgule flottante
- Bits réservés

Chaque sous-champ est identifié par un masque de bit qui a 1 bit dans les positions correspondant au sous-champ, et 0 bit ailleurs. Les bits sont numérotés de sorte que le bit 0 est le bit de poids fort, et le bit 31 le bit de poids faible. Les masques suivants sont définis:

**MQENC\_INTEGER\_MASK**

Masque pour le codage d'entier binaire.

Cette sous-zone occupe les positions de bit 28 à 31 dans la zone *Encoding* .

**MQENC\_DECIMAL\_MASK**

Masque pour le codage d'entier décimal condensé.

Cette sous-zone occupe les positions de bit 24 à 27 dans la zone *Encoding* .

**MQENC\_FLOAT\_MASK**

Masque pour le codage en virgule flottante.

Cette sous-zone occupe les positions de bit 20 à 23 dans la zone *Encoding* .

**MASQUE\_RÉSERVÉ\_MQENC\_MASQUE**

Masque pour les bits réservés.

Cette sous-zone occupe les positions de bit 0 à 19 dans la zone *Encoding* .

## Codage d'entiers binaires



Les valeurs suivantes sont valides pour le codage d'entier binaire:

#### **MQENC\_INTEGER\_UNDEFINI**

Les entiers binaires sont représentés à l'aide d'un codage non défini.

#### **MQENC\_INTEGER\_NORMAL**

Les entiers binaires sont représentés de manière classique:

- L'octet le moins significatif du nombre a l'adresse la plus élevée de tous les octets du nombre ; l'octet le plus significatif a l'adresse la plus faible
- Le bit de poids faible de chaque octet est adjacent à l'octet avec l'adresse supérieure suivante ; le bit de poids fort de chaque octet est adjacent à l'octet avec l'adresse inférieure suivante

#### **MQENC\_INTEGER\_REVERSED**

Les entiers binaires sont représentés de la même manière que MQENC\_INTEGER\_NORMAL, mais avec les octets disposés dans l'ordre inverse. Les bits de chaque octet sont organisés de la même manière que MQENC\_INTEGER\_NORMAL.

## **Codage d'entier décimal condensé**

Les valeurs suivantes sont valides pour le codage d'entier décimal condensé:

#### **MQENC\_DECIMAL\_UNDEFINI**

Les entiers décimaux condensés sont représentés à l'aide d'un codage non défini.

#### **MQENC\_DECIMAL\_NORMAL**

Les entiers décimaux condensés sont représentés de manière conventionnelle:

- Chaque chiffre décimal dans la forme imprimable du nombre est représenté en décimal condensé par un chiffre hexadécimal unique compris entre X' 0' et X' 9'. Chaque chiffre hexadécimal occupe quatre bits, de sorte que chaque octet du nombre décimal condensé représente deux chiffres décimaux dans la forme imprimable du nombre.
- L'octet le moins significatif dans le nombre décimal condensé est l'octet qui contient le chiffre décimal le moins significatif. Dans cet octet, les quatre bits de poids fort contiennent le chiffre décimal de poids faible, et les quatre bits de poids faible contiennent le signe. Le signe est soit X'C'(positif), X'D'(négatif), soit X'F'(non signé).
- L'octet le moins significatif du nombre a l'adresse la plus élevée de tous les octets du nombre ; l'octet le plus significatif a l'adresse la plus faible.
- Le bit de poids faible de chaque octet est adjacent à l'octet avec l'adresse supérieure suivante ; le bit de poids fort de chaque octet est adjacent à l'octet avec l'adresse inférieure suivante.

#### **MQENC\_DECIMAL\_REVERSED**

Les entiers décimaux condensés sont représentés de la même manière que MQENC\_DECIMAL\_NORMAL, mais avec les octets organisés dans l'ordre inverse. Les bits de chaque octet sont organisés de la même manière que MQENC\_DECIMAL\_NORMAL.

## **Codage à virgule flottante**

Les valeurs admises pour le codage en virgule flottante sont les suivantes:

#### **MQENC\_FLOAT\_NON DEFINI**

Les nombres à virgule flottante sont représentés à l'aide d'un codage non défini.

#### **MQENC\_FLOAT\_IEEE\_NORMAL**

Les nombres à virgule flottante sont représentés à l'aide de la norme IEEE<sup>4</sup> format à virgule flottante, avec les octets organisés comme suit:

- L'octet de poids faible de la mantisse a l'adresse la plus élevée de tous les octets du nombre ; l'octet contenant l'exposant a l'adresse la plus faible

---

<sup>4</sup> L'Institut des ingénieurs électriciens et électroniciens

- Le bit de poids faible de chaque octet est adjacent à l'octet avec l'adresse supérieure suivante ; le bit de poids fort de chaque octet est adjacent à l'octet avec l'adresse inférieure suivante

Les détails du codage IEEE float se trouvent dans la norme IEEE 754.

### **MQENC\_FLOAT\_IEEE\_REVERSED**

Les nombres à virgule flottante sont représentés de la même manière que MQENC\_FLOAT\_IEEE\_NORMAL, mais avec les octets organisés dans l'ordre inverse. Les bits de chaque octet sont organisés de la même manière que MQENC\_FLOAT\_IEEE\_NORMAL.

### **MQENC\_FLOAT\_S390**

Les nombres à virgule flottante sont représentés à l'aide du format à virgule flottante System/390 standard ; ce format est également utilisé par System/370.

## **Construction de codages**

Pour construire une valeur pour la zone *Encoding* dans MQMD, les constantes appropriées qui décrivent les codages requis peuvent être ajoutées ensemble (n'ajoutez pas la même constante plusieurs fois) ou combinées à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations bit).

Quelle que soit la méthode utilisée, combinez un seul des codages MQENC\_INTEGER\_\* avec l'un des codages MQENC\_DECIMAL\_\* et l'un des codages MQENC\_FLOAT\_\*.

## **Analyse des codages**

La zone *Encoding* contient des sous-zones ; pour cette raison, les applications qui doivent examiner le codage de type entier, décimal condensé ou flottant doivent utiliser l'une des techniques décrites.

## **Utilisation d'opérations de bit**

Si le langage de programmation prend en charge les opérations de bit, procédez comme suit:

1. Sélectionnez l'une des valeurs suivantes, en fonction du type de codage requis:

- MQENC\_INTEGER\_MASK pour le codage d'entier binaire
- MQENC\_DECIMAL\_MASK pour le codage d'entier décimal condensé
- MQENC\_FLOAT\_MASK pour le codage en virgule flottante

Appelez la valeur A.

2. Combinez la zone *Encoding* avec A à l'aide de l'opération AND bit à bit ; appelez le résultat B.
3. B est le codage requis et peut être testé pour l'égalité avec chacune des valeurs valides pour ce type de codage.

## **Utilisation de l'arithmétique**

Si le langage de programmation *ne prend pas* en charge les opérations bit, effectuez les étapes suivantes à l'aide de l'arithmétique entière:

1. Sélectionnez l'une des valeurs suivantes, en fonction du type de codage requis:

- 1 pour le codage d'entier binaire
- 16 pour le codage d'entier décimal condensé
- 256 pour le codage à virgule flottante

Appelez la valeur A.

2. Divisez la valeur de la zone *Encoding* par A ; appelez le résultat B.
3. Divisez B par 16 ; appelez le résultat C.
4. Multipliez C par 16 et soustrayez de B ; appelez le résultat D.
5. Multipliez D par A ; appelez le résultat E.

6. E est le codage requis et peut être testé pour l'égalité avec chacune des valeurs valides pour ce type de codage.

## Récapitulatif des codages de l'architecture de la machine

Les codages des architectures de machine sont présentés dans la [Tableau 631](#), à la page 947.

Architecture de la machine	Codage d'entier binaire	Codage d'entier décimal condensé	Codage en virgule flottante
IBM i	normal	normal	IEEE normal
Intel x86	reversed	reversed	IEEE inversé
PowerPC	normal	normal	IEEE normal
System/390	normal	normal	System/390

## Options de rapport et indicateurs de message

Cette section décrit les zones *Report* et *MsgFlags* qui font partie du descripteur de message MQMD spécifié dans les appels MQGET, MQPUT et MQPUT1.

Les rubriques de cette section décrivent:

- Structure de la zone de rapport et mode de traitement du gestionnaire de files d'attente
- Comment une application analyse la zone de rapport
- Structure de la zone d'indicateurs de message

Pour plus d'informations sur le descripteur de message MQMD, voir «MQMD-Descripteur de message», à la page 438.

## Structure de la zone de rapport

Ces informations décrivent la structure de la zone de rapport.

La zone *Report* est un entier 32 bits qui est divisé en trois sous-zones distinctes. Ces sous-zones identifient:

- Options de rapport rejetées si le gestionnaire de files d'attente local ne les reconnaît pas
- Options de rapport toujours acceptées, même si le gestionnaire de files d'attente local ne les reconnaît pas
- Options de rapport acceptées uniquement si certaines autres conditions sont remplies

Chaque sous-champ est identifié par un masque de bit qui a 1 bit dans les positions correspondant au sous-champ, et 0 bit ailleurs. Les bits d'un sous-champ ne sont pas nécessairement adjacents. Les bits sont numérotés de sorte que le bit 0 est le bit de poids fort, et le bit 31 le bit de poids faible. Les masques suivants sont définis pour identifier les sous-zones:

### MQRO\_REJECT\_UNSUP\_MASK

Ce masque identifie les positions de bit dans la zone *Report* où les options de rapport qui ne sont pas prises en charge par le gestionnaire de files d'attente local entraînent l'échec de l'appel MQPUT ou MQPUT1 avec le code achèvement MQCC\_FAILED et le code anomalie MQRC\_REPORT\_OPTIONS\_ERROR.

Cette sous-zone occupe les positions de bit 3, et 11 à 13.

### MQRO\_ACCEPT\_UNSUP\_MASK

Ce masque identifie les positions de bit dans la zone *Report* où les options de rapport qui ne sont pas prises en charge par le gestionnaire de files d'attente local sont néanmoins acceptées

sur les appels MQPUT ou MQPUT1 . Le code achèvement MQCC\_WARNING avec le code anomalie MQRC\_UNKNOWN\_REPORT\_OPTION est renvoyé dans ce cas.

Cette sous-zone occupe les positions de bit 0 à 2, 4 à 10 et 24 à 31.

Les options de rapport suivantes sont incluses dans cette sous-zone:

- ACTIVITE MQRO
- MQRO\_COPY\_MSG\_ID\_TO\_CORREL\_ID
- MQRO\_DEAD\_LETTER\_Q
- MQRO\_DISCARD\_MSG
- MQRO\_EXCEPTION
- MQRO\_EXCEPTION\_WITH\_DATA
- MQRO\_EXCEPTION\_WITH\_FULL\_DATA
- MQRO\_EXPIRATION
- MQRO\_EXPIRATION\_WITH\_DATA
- MQRO\_EXPIRATION\_WITH\_FULL\_DATA
- MQRO\_NAN
- MQRO\_NEW\_MSG\_ID
- MQRO\_AUCUN
- MQRO\_PAN
- MQRO\_PASS\_CORREL\_ID
- MQRO\_PASS\_MSG\_ID

#### **MQRO\_ACCEPT\_UNSUP\_IF\_XMIT\_MASK**

Ce masque identifie les positions de bit dans la zone *Report* où les options de rapport qui ne sont pas prises en charge par le gestionnaire de files d'attente local sont néanmoins acceptées sur les appels MQPUT ou MQPUT1 à condition que les deux conditions suivantes soient remplies:

- Le message est destiné à un gestionnaire de files d'attente éloignées.
- L'application ne place pas le message directement dans une file d'attente de transmission locale (c'est-à-dire que la file d'attente identifiée par les zones *ObjectQMgrName* et *ObjectName* dans le descripteur d'objet spécifié dans l'appel MQOPEN ou MQPUT1 n'est pas une file d'attente de transmission locale).

Le code achèvement MQCC\_WARNING avec le code anomalie MQRC\_UNKNOWN\_REPORT\_OPTION est renvoyé si ces conditions sont satisfaites et MQCC\_FAILED avec le code anomalie MQRC\_REPORT\_OPTIONS\_ERROR si ce n'est pas le cas.

Cette sous-zone occupe les positions de bit 14 à 23.

Les options de rapport suivantes sont incluses dans cette sous-zone:

- MQRO\_COA
- MQRO\_COA\_WITH\_DATA
- MQRO\_COA\_WITH\_FULL\_DATA
- MQRO\_COD
- MQRO\_COD\_WITH\_DATA
- MQRO\_COD\_WITH\_FULL\_DATA

Si des options sont spécifiées dans la zone *Report* que le gestionnaire de files d'attente ne reconnaît pas, le gestionnaire de files d'attente vérifie chaque sous-zone à tour de rôle en utilisant l'opération bit à bit AND pour combiner la zone *Report* avec le masque de cette sous-zone. Si le résultat de cette opération est différent de zéro, le code achèvement et les codes raison décrits précédemment sont renvoyés.

Si MQCC\_WARNING est renvoyé, il n'est pas défini quel code anomalie est renvoyé si d'autres conditions d'avertissement existent.

La possibilité de spécifier et d'accepter des options de rapport qui ne sont pas reconnues par le gestionnaire de files d'attente local est utile lors de l'envoi d'un message avec une option de rapport qui est reconnue et traitée par un gestionnaire de files d'attente *distant*.

## Analyse de la zone de rapport

La zone *Report* contient des sous-zones. Par conséquent, les applications qui doivent vérifier si l'expéditeur du message a demandé un rapport particulier doivent utiliser l'une des techniques décrites.

## Utilisation d'opérations de bit

Si le langage de programmation prend en charge les opérations de bit, procédez comme suit:

1. Sélectionnez l'une des valeurs suivantes, en fonction du type de rapport à vérifier:

- MQRO\_COA\_WITH\_FULL\_DATA pour le rapport COA
- MQRO\_COD\_WITH\_FULL\_DATA pour le rapport COD
- MQRO\_EXCEPTION\_WITH\_FULL\_DATA pour le rapport d'exception
- MQRO\_EXPIRATION\_WITH\_FULL\_DATA pour le rapport d'expiration

Appelez la valeur A.

Sous z/OS, utilisez les valeurs MQRO\_\*\_WITH\_DATA à la place des valeurs MQRO\_\*\_WITH\_FULL\_DATA.

2. Combinez la zone *Report* avec A à l'aide de l'opération AND bit à bit ; appelez le résultat B.

3. Testez B pour l'égalité avec chaque valeur possible pour ce type de rapport.

Par exemple, si A est MQRO\_EXCEPTION\_WITH\_FULL\_DATA, testez B pour l'égalité avec chacun des éléments suivants afin de déterminer ce qui a été spécifié par l'expéditeur du message:

- MQRO\_AUCUN
- MQRO\_EXCEPTION
- MQRO\_EXCEPTION\_WITH\_DATA
- MQRO\_EXCEPTION\_WITH\_FULL\_DATA

Les tests peuvent être effectués dans l'ordre qui convient le mieux à la logique de l'application.

Utilisez une méthode similaire pour tester les options MQRO\_PASS\_MSG\_ID ou MQRO\_PASS\_CORREL\_ID ; sélectionnez comme valeur A la valeur appropriée de ces deux constantes, puis procédez comme indiqué précédemment.

## Utilisation de l'arithmétique

Si le langage de programmation *ne prend pas* en charge les opérations bit, effectuez les étapes suivantes à l'aide de l'arithmétique entière:

1. Sélectionnez l'une des valeurs suivantes, en fonction du type de rapport à vérifier:

- Rapport MQRO\_COA for COA
- Rapport MQRO\_COD pour COD
- MQRO\_EXCEPTION pour le rapport d'exception
- MQRO\_EXPIRATION pour le rapport d'expiration

Appelez la valeur A.

2. Divisez la zone *Report* par A ; appeler le résultat B.

3. Divisez B par 8 ; appeler le résultat C.

4. Multipliez C par 8 et soustrayez de B ; appeler le résultat D.
5. Multipliez D par A ; appeler le résultat E.
6. Testez E pour l'égalité avec chaque valeur possible pour ce type de rapport.

Par exemple, si A est MQRO\_EXCEPTION, testez E pour l'égalité avec chacun des éléments suivants afin de déterminer ce qui a été spécifié par l'expéditeur du message:

- MQRO\_AUCUN
- MQRO\_EXCEPTION
- MQRO\_EXCEPTION\_WITH\_DATA
- MQRO\_EXCEPTION\_WITH\_FULL\_DATA

Les tests peuvent être effectués dans l'ordre qui convient le mieux à la logique de l'application.

Le pseudocode suivant illustre cette technique pour les messages de rapport d'exception:

```
A = MQRO_EXCEPTION
B = Report/A
C = B/8
D = B - C*8
E = D*A
```

Utilisez une méthode similaire pour tester les options MQRO\_PASS\_MSG\_ID ou MQRO\_PASS\_CORREL\_ID ; sélectionnez comme valeur A la valeur appropriée de ces deux constantes, puis procédez comme indiqué précédemment, mais remplacez la valeur 8 dans les étapes précédentes par la valeur 2.

## Structure de la zone d'indicateurs de message

Ces informations décrivent la structure de la zone d'indicateurs de message.

La zone *MsgFlags* est un entier 32 bits qui est divisé en trois sous-zones distinctes. Ces sous-zones identifient:

- Indicateurs de message rejetés si le gestionnaire de files d'attente local ne les reconnaît pas
- Indicateurs de message toujours acceptés, même si le gestionnaire de files d'attente local ne les reconnaît pas
- Indicateurs de message acceptés uniquement si certaines autres conditions sont remplies

**Remarque :** Toutes les sous-zones de *MsgFlags* sont réservées pour être utilisées par le gestionnaire de files d'attente.

Chaque sous-champ est identifié par un masque de bit qui a 1 bit dans les positions correspondant au sous-champ, et 0 bit ailleurs. Les bits sont numérotés de sorte que le bit 0 est le bit de poids fort, et le bit 31 le bit de poids faible. Les masques suivants sont définis pour identifier les sous-zones:

### MQMF\_REJECT\_UNSUP\_MASK

Ce masque identifie les positions de bit dans la zone *MsgFlags* où les indicateurs de message qui ne sont pas pris en charge par le gestionnaire de files d'attente local entraînent l'échec de l'appel MQPUT ou MQPUT1 avec le code achèvement MQCC\_FAILED et le code anomalie MQRC\_MSG\_FLAGS\_ERROR.

Cette sous-zone occupe les positions de bit 20 à 31.

Les indicateurs de message suivants sont inclus dans cette sous-zone:

- MQMF\_LAST\_MSG\_IN\_GROUP
- SEGMENT\_DERNIER\_MQMF\_SEGMENT
- GROUPE MQMF\_MSG\_IN\_GROUP
- SEGMENT\_MQMF
- MQMF\_SEGMENTATION\_ALLOWED

- MQMF\_SEGMENTATION\_INHIBÉE

### **MQMF\_ACCEPT\_UNSUP\_MASK**

Ce masque identifie les positions de bit dans la zone *MsgFlags* où les indicateurs de message qui ne sont pas pris en charge par le gestionnaire de files d'attente local sont néanmoins acceptés sur les appels MQPUT ou MQPUT1. Le code achèvement est MQCC\_OK.

Cette sous-zone occupe les positions de bit 0 à 11.

### **MQMF\_ACCEPT\_UNSUP\_IF\_XMIT\_MASK**

Ce masque identifie les positions de bit dans la zone *MsgFlags* où les indicateurs de message qui ne sont pas pris en charge par le gestionnaire de files d'attente local sont néanmoins acceptés sur les appels MQPUT ou MQPUT1 à condition que les deux conditions suivantes soient satisfaites:

- Le message est destiné à un gestionnaire de files d'attente éloignées.
- L'application ne place pas le message directement dans une file d'attente de transmission locale (c'est-à-dire que la file d'attente identifiée par les zones *ObjectQMGrName* et *ObjectName* dans le descripteur d'objet spécifié dans l'appel MQOPEN ou MQPUT1 n'est pas une file d'attente de transmission locale).

Le code achèvement MQCC\_OK est renvoyé si ces conditions sont satisfaites et MQCC\_FAILED avec le code anomalie MQRC\_MSG\_FLAGS\_ERROR si ce n'est pas le cas.

Cette sous-zone occupe les positions de bit 12 à 19.

Si des indicateurs sont spécifiés dans la zone *MsgFlags* que le gestionnaire de files d'attente ne reconnaît pas, le gestionnaire de files d'attente vérifie chaque sous-zone à tour de rôle en utilisant l'opération bit à bit AND pour combiner la zone *MsgFlags* avec le masque de cette sous-zone. Si le résultat de cette opération est différent de zéro, le code achèvement et les codes raison décrits précédemment sont renvoyés.

## **Exit de conversion de données**

Cette collection de rubriques décrit l'interface de l'exit de conversion de données et le traitement effectué par le gestionnaire de files d'attente lorsque la conversion de données est requise.

Pour plus d'informations sur la conversion de données, voir *Data Conversion under IBM MQ* à l'adresse <https://www.ibm.com/support/pages/node/317869>.

L'exit de conversion de données est appelé dans le cadre du traitement de l'appel MQGET afin de convertir les données de message d'application à la représentation requise par l'application de réception. La conversion des données de message d'application est facultative ; elle nécessite que l'option MQGMO\_CONVERT soit spécifiée sur l'appel MQGET.

Les sujets suivants sont décrits:

- Traitement effectué par le gestionnaire de files d'attente en réponse à l'option MQGMO\_CONVERT ; voir «[Traitement de la conversion](#)», à la page 952.
- Conventions de traitement utilisées par le gestionnaire de files d'attente lors du traitement d'un format intégré ; ces conventions sont également recommandées pour les exits écrits par l'utilisateur. Voir «[Conventions de traitement](#)», à la page 953.
- Remarques spéciales relatives à la conversion des messages de rapport ; voir «[Conversion des messages de rapport](#)», à la page 957.
- Les paramètres transmis à l'exit de conversion de données ; voir «[MQ\\_DATA\\_CONV\\_EXIT-Exit de conversion de données](#)», à la page 970.
- Appel pouvant être utilisé à partir de l'exit pour convertir des données de type caractères entre différentes représentations ; voir «[MQXCNCV-Conversion de caractères](#)», à la page 964.
- Paramètre de structure de données spécifique à l'exit ; voir «[MQDXP-Paramètre d'exit de conversion de données](#)», à la page 958.

## Traitement de la conversion

Ces informations décrivent le traitement effectué par le gestionnaire de files d'attente en réponse à l'option MQGMO\_CONVERT.

Le gestionnaire de files d'attente effectue les actions suivantes si l'option MQGMO\_CONVERT est spécifiée dans l'appel MQGET et qu'un message doit être renvoyé à l'application:

1. Si une ou plusieurs des conditions suivantes sont remplies, aucune conversion n'est nécessaire:
  - Les données de message se trouvent déjà dans le jeu de caractères et le codage requis par l'application émettant l'appel MQGET. L'application doit définir les zones *CodedCharSetId* et *Encoding* dans le paramètre **MsgDesc** de l'appel MQGET sur les valeurs requises, avant d'émettre l'appel.
  - La longueur des données de message est égale à zéro.
  - La longueur du paramètre **Buffer** de l'appel MQGET est égale à zéro.

Dans ces cas, le message est renvoyé sans conversion à l'application émettant l'appel MQGET ; les valeurs *CodedCharSetId* et *Encoding* du paramètre **MsgDesc** sont définies sur les valeurs des informations de contrôle du message et l'appel se termine avec l'une des combinaisons suivantes de code achèvement et de code anomalie:

Tableau 632. Combinaisons de codes achèvement et de codes raison

Code de fin d'exécution	Code raison
MQCC_OK	MQRC_NONE
MQCC_WARNING	MQRC_TRUNCATED_MSG_ACCEPTED
MQCC_WARNING	MQRC_TRUNCATED_MSG_FAILED

Les étapes suivantes sont effectuées uniquement si le jeu de caractères ou le codage des données de message diffère de la valeur correspondante dans le paramètre **MsgDesc** et que des données doivent être converties:

2. Si la zone *Format* des informations de contrôle du message a la valeur MQFMT\_NONE, le message est renvoyé non converti avec le code achèvement MQCC\_WARNING et le code anomalie MQRC\_FORMAT\_ERROR.

Dans tous les autres cas, le traitement de la conversion se poursuit.

3. Le message est supprimé de la file d'attente et placé dans une mémoire tampon temporaire de même taille que le paramètre **Buffer**. Pour les opérations de navigation, le message est copié dans la mémoire tampon temporaire au lieu d'être supprimé de la file d'attente.
4. Si le message doit être tronqué pour tenir dans la mémoire tampon, les opérations suivantes sont effectuées:
  - Si l'option MQGMO\_ACCEPT\_TRUNCATED\_MSG n'a pas été spécifiée, le message est renvoyé non converti avec le code achèvement MQCC\_WARNING et le code anomalie MQRC\_TRUNCATED\_MSG\_FAILED.
  - Si l'option MQGMO\_ACCEPT\_TRUNCATED\_MSG a été spécifiée, le code achèvement est défini sur MQCC\_WARNING, le code anomalie est défini sur MQRC\_TRUNCATED\_MSG\_ACCEPTED et le traitement de la conversion se poursuit.
5. Si le message peut être hébergé dans la mémoire tampon sans troncature ou si l'option MQGMO\_ACCEPT\_TRUNCATED\_MSG a été spécifiée, les opérations suivantes sont effectuées:
  - Si le format est un format intégré, la mémoire tampon est transmise au service de conversion de données du gestionnaire de files d'attente.
  - Si le format n'est pas un format intégré, la mémoire tampon est transmise à un exit écrit par l'utilisateur portant le même nom que le format. Si l'exit est introuvable, le message est renvoyé sans conversion, avec le code achèvement MQCC\_WARNING et le code anomalie MQRC\_FORMAT\_ERROR.



Si aucune erreur ne se produit, la sortie du service de conversion de données ou de l'exit écrit par l'utilisateur est le message converti, ainsi que le code achèvement et le code raison à renvoyer à l'application émettant l'appel MQGET.

6. Si la conversion aboutit, le gestionnaire de files d'attente renvoie le message converti à l'application. Dans ce cas, le code achèvement et le code anomalie renvoyés par l'appel MQGET sont l'une des combinaisons suivantes:

Tableau 633. Combinaisons de codes achèvement et de codes raison

Code de fin d'exécution	Code raison
MQCC_OK	MQRC_NONE
MQCC_WARNING	MQRC_TRUNCATED_MSG_ACCEPTED

Toutefois, si la conversion est effectuée par un exit écrit par l'utilisateur, d'autres codes raison peuvent être renvoyés, même lorsque la conversion aboutit.

Si la conversion échoue, le gestionnaire de files d'attente renvoie le message non converti à l'application, avec les zones *CodedCharSetId* et *Encoding* du paramètre **MsgDesc** définies sur les valeurs des informations de contrôle du message, et avec le code achèvement MQCC\_WARNING.

## Conventions de traitement

Lors de la conversion d'un format intégré, le gestionnaire de files d'attente respecte les conventions de traitement décrites.

Les exits écrits par l'utilisateur doivent également respecter ces conventions, bien que cela ne soit pas appliqué par le gestionnaire de files d'attente. Les formats intégrés convertis par le gestionnaire de files d'attente sont les suivants:

- MQFMT\_ADMIN
- MQFMT\_CICS (z/OS uniquement)
- MQFMT\_COMMAND\_1
- MQFMT\_COMMAND\_2
- MQFMT\_DEAD\_LETTER\_HEADER
- MQFMT\_DIST\_HEADER
- MQFMT\_EVENT version 1
- MQFMT\_EVENT version 2
- MQFMT\_IMS
- MQFMT\_IMS\_VAR\_STRING
- MQFMT\_MD\_EXTENSION
- MQFMT\_PCF
- MQFMT\_REF\_MSG\_HEADER
- MQFMT\_RF\_HEADER
- MQFMT\_RF\_HEADER\_2
- MQFMT\_STRING
- MQFMT\_TRIGGER
- MQFMT\_WORK\_INFO\_HEADER (z/OS uniquement)
- MQFMT\_XMIT\_Q\_HEADER

1. Si le message se développe lors de la conversion et dépasse la taille du paramètre **Buffer**, procédez comme suit:

- Si l'option MQGMO\_ACCEPT\_TRUNCATED\_MSG n'a pas été spécifiée, le message est renvoyé non converti avec le code achèvement MQCC\_WARNING et le code anomalie MQRC\_CONVERTED\_MSG\_TOO\_BIG.
  - Si l'option MQGMO\_ACCEPT\_TRUNCATED\_MSG a été spécifiée, le message est tronqué, le code achèvement est défini sur MQCC\_WARNING, le code anomalie est défini sur MQRC\_TRUNCATED\_MSG\_ACCEPTED et le traitement de la conversion se poursuit.
2. En cas de troncature (avant ou pendant la conversion), le nombre d'octets valides renvoyés dans le paramètre **Buffer** peut être inférieur à la longueur de la mémoire tampon.
- Cela peut se produire, par exemple, si un entier de 4 octets ou un caractère DBCS chevauche la fin de la mémoire tampon. L'élément incomplet des informations n'est pas converti et les octets du message renvoyé ne contiennent pas d'informations valides. Cela peut également se produire si un message tronqué avant la conversion est réduit lors de la conversion.
- Si le nombre d'octets valides renvoyés est inférieur à la longueur de la mémoire tampon, les octets inutilisés à la fin de la mémoire tampon sont définis sur null.
3. Si un tableau ou une chaîne chevauche la fin de la mémoire tampon, autant de données que possible sont converties ; seul l'élément de tableau ou le caractère DBCS qui est incomplet n'est pas converti ; les éléments de tableau ou les caractères précédents sont convertis.
4. En cas de troncature (avant ou pendant la conversion), la longueur renvoyée pour le paramètre **DataLength** correspond à la longueur du message non converti avant la troncature.
5. Lorsque des chaînes sont converties entre des jeux de caractères codés sur un octet (SBCS), des jeux de caractères codés sur deux octets (DBCS) ou des jeux de caractères codés sur plusieurs octets (MBCS), les chaînes peuvent se développer ou se contracter.
- Dans les formats PCF MQFMT\_ADMIN, MQFMT\_EVENT et MQFMT\_PCF, les chaînes des structures MQCFST et MQCFSL se développent ou se contractent selon les besoins pour s'adapter à la chaîne après la conversion.
- Pour la structure de liste de chaînes MQCFSL, les chaînes de la liste peuvent se développer ou se contracter différemment. Dans ce cas, le gestionnaire de files d'attente remplit les chaînes plus courtes avec des blancs pour qu'elles aient la même longueur que la chaîne la plus longue après la conversion.
- Dans le format MQFMT\_REF\_MSG\_HEADER, les chaînes adressées par les zones SrcEnvOffset, SrcNameOffset, DestEnvOffset et DestNameOffset se développent ou se contractent selon les besoins pour s'adapter aux chaînes après la conversion.
  - Dans le format MQFMT\_RF\_HEADER, la zone NameValueString se développe ou se contracte comme nécessaire pour s'adapter aux paires nom-valeur après la conversion.
  - Dans les structures avec des tailles de champ fixes, le gestionnaire de files d'attente permet aux chaînes de se développer ou de se contracter dans leurs champs fixes, à condition qu'aucune information significative ne soit perdue. A cet égard, les blancs de fin et les caractères qui suivent le premier caractère nul de la zone sont considérés comme non significatifs.
    - Si la chaîne se développe, mais que seuls des caractères non significatifs doivent être supprimés pour prendre en charge la chaîne convertie dans la zone, la conversion aboutit et l'appel se termine avec MQCC\_OK et le code anomalie MQRC\_NONE (en supposant qu'il n'y ait pas d'autres erreurs).
    - Si la chaîne se développe, mais que la chaîne convertie nécessite que des caractères significatifs soient supprimés pour tenir dans la zone, le message est renvoyé non converti et l'appel se termine avec MQCC\_WARNING et le code anomalie MQRC\_CONVERTED\_STRING\_TOO\_BIG.

**Remarque :** Le code anomalie MQRC\_CONVERTED\_STRING\_TOO\_BIG indique dans ce cas si l'option MQGMO\_ACCEPT\_TRUNCATED\_MSG a été spécifiée ou non.

    - Si la chaîne se contracte, le gestionnaire de files d'attente complète la chaîne avec des blancs à la longueur de la zone.
6. Pour les messages comportant une ou plusieurs structures d'en-tête MQ suivies de données utilisateur, une ou plusieurs structures d'en-tête peuvent être converties, alors que le reste du

message ne l'est pas. Toutefois, à deux exceptions près, les zones *CodedCharSetId* et *Encoding* de chaque structure d'en-tête indiquent toujours correctement le jeu de caractères et le codage des données qui suivent la structure d'en-tête.

Les deux exceptions sont les structures MQCIH et MQIIH, où les valeurs des zones *CodedCharSetId* et *Encoding* de ces structures ne sont pas significatives. Pour ces structures, les données qui suivent la structure se trouvent dans le même jeu de caractères et le même codage que la structure MQCIH ou MQIIH elle-même.

7. Si les zones *CodedCharSetId* ou *Encoding* dans les informations de contrôle du message en cours d'extraction ou dans le paramètre **MsgDesc** indiquent des valeurs non définies ou non prises en charge, le gestionnaire de files d'attente peut ignorer l'erreur si la valeur non définie ou non prise en charge n'a pas besoin d'être utilisée pour convertir le message.

Par exemple, si la zone *Encoding* du message spécifie un codage à virgule flottante non pris en charge, mais que le message contient uniquement des données entières ou des données à virgule flottante qui ne nécessitent pas de conversion (car les codages à virgule flottante source et cible sont identiques), l'erreur peut ne pas être diagnostiquée.

Si l'erreur est diagnostiquée, le message est renvoyé sans conversion, avec le code achèvement MQCC\_WARNING et l'un des codes anomalie MQRC\_SOURCE\_\*\_ERROR ou MQRC\_TARGET\_\*\_ERROR (selon le cas) ; les zones *CodedCharSetId* et *Encoding* du paramètre **MsgDesc** sont définies sur les valeurs des informations de contrôle du message.

Si l'erreur n'est pas diagnostiquée et que la conversion aboutit, les valeurs renvoyées dans les zones *CodedCharSetId* et *Encoding* du paramètre **MsgDesc** sont celles spécifiées par l'application émettrice de l'appel MQGET.

8. Dans tous les cas, si le message est renvoyé à l'application non convertie, le code achèvement est défini sur MQCC\_WARNING et les zones *CodedCharSetId* et *Encoding* du paramètre **MsgDesc** sont définies sur les valeurs appropriées aux données non converties. Cette opération est également effectuée pour MQFMT\_NONE.

Le paramètre **Reason** est défini sur un code qui indique la raison pour laquelle la conversion n'a pas pu être effectuée, sauf si le message a également dû être tronqué ; les codes raison liés à la troncature sont prioritaires sur les codes raison liés à la conversion. (Pour déterminer si un message tronqué a été converti, vérifiez les valeurs renvoyées dans les zones *CodedCharSetId* et *Encoding* du paramètre **MsgDesc**.)

Lorsqu'une erreur est diagnostiquée, un code anomalie spécifique est renvoyé ou le code anomalie général MQRC\_NOT\_CONVERTIE. Le code anomalie renvoyé dépend des capacités de diagnostic du service de conversion de données sous-jacent.

9. Si le code achèvement MQCC\_WARNING est renvoyé et que plusieurs codes anomalie sont pertinents, l'ordre de priorité est le suivant:
  - a. Les raisons suivantes prévalent sur toutes les autres ; une seule des raisons de ce groupe peut se produire:
    - MQRC\_SIGNAL\_REQUEST\_ACCEPTED
    - MQRC\_TRUNCATED\_MSG\_ACCEPTED
  - b. L'ordre de priorité dans les codes anomalie restants n'est pas défini.

10. A la fin de l'appel MQGET:

- Le code anomalie suivant indique que la conversion du message a abouti:
  - MQRC\_NONE
- Les codes anomalie suivants indiquent que le message *peut* avoir été converti (consultez les zones *CodedCharSetId* et *Encoding* du paramètre **MsgDesc** pour le savoir):
  - MQRC\_MSG\_MARKED\_BROWSE\_CO\_OP
  - MQRC\_TRUNCATED\_MSG\_ACCEPTED
- Tous les autres codes anomalie indiquent que le message n'a pas été converti.

Le traitement suivant est spécifique aux formats intégrés ; il ne s'applique pas aux formats définis par l'utilisateur:

11. A l'exception des formats suivants:

- MQFMT\_ADMIN
- MQFMT\_COMMAND\_1
- MQFMT\_COMMAND\_2
- MQFMT\_EVENT
- MQFMT\_IMS\_VAR\_STRING
- MQFMT\_PCF
- MQFMT\_STRING

aucun des formats intégrés ne peut être converti depuis ou vers des jeux de caractères qui ne comportent pas de caractères SBCS pour les caractères admis dans les noms de file d'attente. Si une tentative est effectuée pour effectuer une telle conversion, le message est renvoyé non converti, avec le code achèvement MQCC\_WARNING et le code anomalie MQRC\_SOURCE\_CCSID\_ERROR ou MQRC\_TARGET\_CCSID\_ERROR, selon le cas.

Le jeu de caractères Unicode UTF-16 est un exemple de jeu de caractères qui ne comporte pas de caractères SBCS pour les caractères valides dans les noms de file d'attente.

12. Si les données de message pour un format intégré sont tronquées, les zones du message qui contiennent des longueurs de chaînes, ou des nombres d'éléments ou de structures, ne sont pas ajustées pour refléter la longueur des données effectivement renvoyées à l'application ; les valeurs renvoyées pour ces zones dans les données de message sont les valeurs applicables au message *avant troncature*.

Lors du traitement de messages tels qu'un message MQFMT\_ADMIN tronqué, assurez-vous que l'application ne tente pas d'accéder aux données au-delà de la fin des données renvoyées.

13. Si le nom de format est MQFMT\_DEAD\_LETTER\_HEADER, les données de message commencent par une structure MQDLH, éventuellement suivie de zéro ou plusieurs octets de données de message d'application. Le format, le jeu de caractères et le codage des données de message d'application sont définis par les zones Format, CodedCharSetId et Encoding de la structure MQDLH au début du message. Etant donné que la structure MQDLH et les données de message d'application peuvent avoir des jeux de caractères et des codages différents, l'un ou l'autre, ou les deux, de la structure MQDLH et des données de message d'application peuvent nécessiter une conversion.

Le gestionnaire de files d'attente convertit d'abord la structure MQDLH, si nécessaire. Si la conversion aboutit ou si la structure MQDLH ne requiert pas de conversion, le gestionnaire de files d'attente vérifie les zones CodedCharSetId et Encoding de la structure MQDLH pour déterminer si la conversion des données de message d'application est requise. Si une conversion est requise, le gestionnaire de files d'attente appelle l'exit écrit par l'utilisateur avec le nom donné par la zone Format dans la structure MQDLH ou effectue la conversion elle-même (si Format est le nom d'un format intégré).

Si l'appel MQGET renvoie le code achèvement MQCC\_WARNING et que le code anomalie indique que la conversion a échoué, l'une des conditions suivantes s'applique:

- La structure MQDLH n'a pas pu être convertie. Dans ce cas, les données de message d'application n'auront pas non plus été converties.
- La structure MQDLH a été convertie, mais les données de message d'application ne l'ont pas été.

L'application peut examiner les valeurs renvoyées dans les zones CodedCharSetId et Encoding du paramètre **MsgDesc**, ainsi que celles de la structure MQDLH, afin de déterminer laquelle des valeurs précédemment appliquées.

14. Si le nom de format est MQFMT\_XMIT\_Q\_HEADER, les données de message commencent par une structure MQXQH, éventuellement suivie de zéro ou plusieurs octets de données supplémentaires. Ces données supplémentaires sont généralement les données de message d'application (qui peuvent

être de longueur nulle), mais il peut également y avoir une ou plusieurs autres structures d'en-tête MQ, au début des données supplémentaires.

La structure MQXQH doit être dans le jeu de caractères et le codage du gestionnaire de files d'attente. Le format, le jeu de caractères et le codage des données qui suivent la structure MQXQH sont fournis par les zones *Format*, *CodedCharSetId* et *Encoding* de la structure MQMD contenue dans MQXQH. Pour chaque structure d'en-tête MQ suivante présente, les zones *Format*, *CodedCharSetId* et *Encoding* de la structure décrivent les données qui suivent cette structure ; ces données sont soit une autre structure d'en-tête MQ, soit les données de message d'application.

Si l'option MQGMO\_CONVERT est spécifiée pour un message MQFMT\_XMIT\_Q\_HEADER, les données de message d'application et certaines des structures d'en-tête MQ sont converties, *mais les données de la structure MQXQH ne sont pas*. En cas de retour de l'appel MQGET, par conséquent:

- Les valeurs des zones *Format*, *CodedCharSetId* et *Encoding* du paramètre **MsgDesc** décrivent les données de la structure MQXQH et non les données de message d'application ; les valeurs ne sont donc pas identiques à celles spécifiées par l'application qui a émis l'appel MQGET.

Cela a pour effet qu'une application qui extrait à plusieurs reprises des messages d'une file d'attente de transmission avec l'option MQGMO\_CONVERT spécifiée doit réinitialiser les zones *CodedCharSetId* et *Encoding* du paramètre **MsgDesc** sur les valeurs requises pour les données de message d'application, avant chaque appel MQGET.

- Les valeurs des zones *Format*, *CodedCharSetId* et *Encoding* de la dernière structure d'en-tête MQ présente décrivent les données du message d'application. Si aucune autre structure d'en-tête MQ n'est présente, les données de message d'application sont décrites par ces zones dans la structure MQMD au sein de la structure MQXQH. Si la conversion aboutit, les valeurs seront identiques à celles spécifiées dans le paramètre **MsgDesc** par l'application qui a émis l'appel MQGET.

Si le message est un message de liste de distribution, la structure MQXQH est suivie d'une structure MQDH (plus ses tableaux d'enregistrements MQOR et MQPMR), qui à son tour peut être suivie de zéro ou plusieurs autres structures d'en-tête MQ et de zéro ou plusieurs octets de données de message d'application. A l'instar de la structure MQXQH, la structure MQDH doit être dans le jeu de caractères et le codage du gestionnaire de files d'attente, et elle n'est pas convertie sur l'appel MQGET, même si l'option MQGMO\_CONVERT est spécifiée.

Le traitement des structures MQXQH et MQDH décrites précédemment est principalement destiné à être utilisé par les agents de canal de message lorsqu'ils extraient des messages des files d'attente de transmission.

## Conversion des messages de rapport

En général, un message de rapport peut contenir des quantités variables de données de message d'application, en fonction des options de rapport spécifiées par l'expéditeur du message d'origine. Toutefois, un rapport d'activité peut contenir des données, mais sans l'option de rapport mentionnant \* \_WITH\_DATA dans la constante.

En particulier, un message de rapport peut contenir:

1. Aucune donnée de message d'application
2. Certaines des données de message d'application du message d'origine

Cela se produit lorsque l'expéditeur du message d'origine spécifie MQRO\_ \* \_WITH\_DATA et que le message a une longueur supérieure à 100 octets.

3. Toutes les données de message d'application du message d'origine

Cela se produit lorsque l'expéditeur du message d'origine spécifie MQRO\_ \* \_WITH\_FULL\_DATA ou MQRO\_ \* \_WITH\_DATA et que le message est de 100 octets ou moins.

Lorsque le gestionnaire de files d'attente ou l'agent MCA génère un message de rapport, il copie le nom de format du message d'origine dans la zone *Format* des informations de contrôle du message de rapport.

Le nom de format dans le message de rapport peut donc impliquer une longueur de données différente de la longueur réellement présente dans le message de rapport (cas 1 et 2 précédemment).

Si l'option MQGMO\_CONVERT est spécifiée lors de l'extraction du message de rapport:

- Pour le cas 1 précédemment, l'exit de conversion de données n'est pas appelé (car le message de rapport ne contient pas de données).
- Pour le cas 3 précédemment, le nom de format correct implique la longueur des données de message.
- Mais pour le cas 2 précédemment, l'exit de conversion de données est appelé pour convertir un message *plus court* que la longueur impliquée par le nom de format.

En outre, le code anomalie transmis à l'exit est généralement MQRC\_NONE (c'est-à-dire que le code anomalie n'indique pas que le message a été tronqué). Cela se produit car les données de message ont été tronquées par l'*émetteur* du message de rapport et non par le gestionnaire de files d'attente du récepteur en réponse à l'appel MQGET.

En raison de ces possibilités, l'exit de conversion de données ne doit pas utiliser le nom de format pour déduire la longueur des données qui lui sont transmises ; il doit plutôt vérifier la longueur des données fournies et être prêt à convertir moins de données que la longueur impliquée par le nom de format. Si les données peuvent être converties correctement, le code achèvement MQCC\_OK et le code anomalie MQRC\_NONE doivent être renvoyés par l'exit. La longueur des données de message à convertir est transmise à l'exit en tant que paramètre **InBufferLength**.

### Interface de programmation sensible au produit

## MQDXP-Paramètre d'exit de conversion de données

La structure MQDXP est un paramètre que le gestionnaire de files d'attente transmet à l'exit de conversion de données lorsque l'exit est appelé pour convertir les données de message dans le cadre du traitement de l'appel MQGET. Pour plus de détails sur l'exit de conversion de données, voir la description de l'appel MQ\_DATA\_CONV\_EXIT.

Les données de type caractère dans MQDXP se trouvent dans le jeu de caractères du gestionnaire de files d'attente local ; elles sont fournies par l'attribut de gestionnaire de files d'attente **CodedCharSetId**. Les données numériques dans MQDXP sont dans le codage de la machine native ; elles sont fournies par MQENC\_NATIVE.

Seules les zones *DataLength*, *CompCode*, *Reason* et *ExitResponse* de MQDXP peuvent être modifiées par l'exit ; les modifications apportées aux autres zones sont ignorées. Toutefois, la zone *DataLength* ne peut pas être modifiée si le message en cours de conversion est un segment qui ne contient qu'une partie d'un message logique.

Lorsque le contrôle revient au gestionnaire de files d'attente à partir de l'exit, le gestionnaire de files d'attente vérifie les valeurs renvoyées dans MQDXP. Si les valeurs renvoyées ne sont pas valides, le gestionnaire de files d'attente poursuit le traitement comme si l'exit avait renvoyé MQXDR\_CONVERSION\_FAILED dans *ExitResponse* ; toutefois, le gestionnaire de files d'attente ignore les valeurs des zones *CompCode* et *Reason* renvoyées par l'exit dans ce cas, et utilise à la place les valeurs que ces zones avaient dans les *entrées* de l'exit. Les valeurs suivantes dans MQDXP entraînent ce traitement:

- Zone *ExitResponse* non MQXDR\_OK et non MQXDR\_CONVERSION\_FAILED
- Zone *CompCode* non MQCC\_OK et non MQCC\_WARNING
- Zone *DataLength* inférieure à zéro ou zone *DataLength* modifiée lorsque le message en cours de conversion est un segment qui ne contient qu'une partie d'un message logique.

Le tableau suivant récapitule les zones de la structure.

Tableau 634. Zones dans MQDXP

<b>Zone</b>	<b>Description</b>	<b>Topic</b>
<i>StrucId</i>	Identificateur de structure	<a href="#">StrucId</a>
<i>Version</i>	Numéro de version de structure	<a href="#">Version</a>
<i>AppOptions</i>	Options d'application	<a href="#">AppOptions</a>
<i>Encoding</i>	Codage numérique requis par l'application	<a href="#">Codage</a>
<i>CodedCharSetId</i>	Jeu de caractères requis par l'application	<a href="#">CodedCharSetId</a>
<i>DataLength</i>	Longueur en octets des données de message	<a href="#">DataLength</a>
<i>CompCode</i>	Code de fin d'exécution	<a href="#">CompCode</a>
<i>Reason</i>	Qualification du code anomalie <i>CompCode</i>	<a href="#">Motif</a>
<i>ExitResponse</i>	Réponse de l'exit	<a href="#">ExitResponse</a>
<i>Hconn</i>	Descripteur de connexion	<a href="#">Nom de connexion</a>
<i>pEntryPoints</i>	Adresse de la structure MQIEP	<a href="#">pEntryPoints</a>

## Zones

La structure MQDXP contient les zones suivantes ; elles sont décrites par ordre alphabétique.

### AppOptions

Type: MQLONG

Il s'agit d'une copie de la zone *Options* de la structure MQGMO spécifiée par l'application émettant l'appel MQGET. L'exit peut avoir besoin de les examiner pour déterminer si l'option MQGMO\_ACCEPT\_TRUNCATED\_MSG a été spécifiée.

Il s'agit d'une zone d'entrée de l'exit.

### CodedCharSetId

Type: MQLONG

Il s'agit de l'identificateur de jeu de caractères codés du jeu de caractères requis par l'application émettant l'appel MQGET ; voir la zone *CodedCharSetId* dans la structure MQMD pour plus de détails. Si l'application spécifie la valeur spéciale MQCCSI\_Q\_MGR dans l'appel MQGET, le gestionnaire de files d'attente remplace cette valeur par l'identificateur de jeu de caractères réel du jeu de caractères utilisé par le gestionnaire de files d'attente, avant d'appeler l'exit.

Si la conversion aboutit, l'exit doit le copier dans la zone *CodedCharSetId* du descripteur de message.

Il s'agit d'une zone d'entrée de l'exit.

### CompCode

Type: MQLONG

Lorsque l'exit est appelé, il contient le code achèvement renvoyé à l'application qui a émis l'appel MQGET, si l'exit ne fait rien. Il s'agit toujours de MQCC\_WARNING, car soit le message a été tronqué, soit le message doit être converti, ce qui n'a pas encore été fait.

Dans la sortie de l'exit, cette zone contient le code achèvement à renvoyer à l'application dans le paramètre **CompCode** de l'appel MQGET ; seuls MQCC\_OK et MQCC\_WARNING sont valides. Consultez

la description de la zone *Reason* pour obtenir des suggestions sur la manière dont l'exit peut définir cette zone sur la sortie.

Il s'agit d'une zone d'entrée-sortie pour l'exit.

### **DataLength**

Type: MQLONG

Lorsque l'exit est appelé, cette zone contient la longueur d'origine des données de message d'application. Si le message a été tronqué pour tenir dans la mémoire tampon fournie par l'application, la taille du message fourni à l'exit est *inférieure* à la valeur de *DataLength*. La taille du message fourni à l'exit est toujours indiquée par le paramètre **InBufferLength** de l'exit, quelle que soit la troncature qui s'est produite.

La troncature est indiquée par la zone *Reason* dont la valeur est MQRC\_TRUNCATED\_MSG\_ACCEPTED en entrée de l'exit.

La plupart des conversions n'ont pas besoin de modifier cette longueur, mais un exit peut le faire si nécessaire ; la valeur définie par l'exit est renvoyée à l'application dans le paramètre **DataLength** de l'appel MQGET. Toutefois, cette longueur ne peut pas être modifiée si le message en cours de conversion est un segment qui ne contient qu'une partie d'un message logique. En effet, si vous modifiez la longueur, les décalages des segments ultérieurs dans le message logique seront incorrects.

Notez que si l'exit souhaite modifier la longueur des données, sachez que le gestionnaire de files d'attente a déjà décidé si les données de message s'intègrent dans la mémoire tampon de l'application, en fonction de la longueur des données *non converties*. Cette décision détermine si le message est supprimé de la file d'attente (ou si le curseur de navigation a été déplacé pour une demande de navigation) et s'il n'est pas affecté par une modification de la longueur des données provoquée par la conversion. Pour cette raison, il est recommandé que les exits de conversion n'entraînent pas de modification de la longueur des données de message d'application.

Si la conversion de caractères implique un changement de longueur, une chaîne peut être convertie en une autre chaîne de même longueur en octets, en tronquant les blancs de fin ou en ajoutant des blancs si nécessaire.

L'exit n'est pas appelé si le message ne contient pas de données de message d'application ; par conséquent, *DataLength* est toujours supérieur à zéro.

Il s'agit d'une zone d'entrée-sortie pour l'exit.

### **Encoding**

Type: MQLONG

Codage numérique requis par l'application.

Il s'agit du codage numérique requis par l'application qui émet l'appel MQGET ; voir la zone *Encoding* dans la structure MQMD pour plus de détails.

Si la conversion aboutit, l'exit le copie dans la zone *Encoding* du descripteur de message.

Il s'agit d'une zone d'entrée de l'exit.

### **ExitOptions**

Type: MQLONG

Il s'agit d'une zone réservée ; sa valeur est 0.

### **ExitResponse**

Type: MQLONG

Réponse de l'exit. Ce paramètre est défini par l'exit pour indiquer la réussite ou non de la conversion. Il doit s'agir de l'une des valeurs suivantes:

#### **MQXDR\_OK**

La conversion a abouti.



Si l'exit spécifie cette valeur, le gestionnaire de files d'attente renvoie la valeur suivante à l'application qui a émis l'appel MQGET:

- Valeur de la zone *CompCode* dans la sortie de l'exit
- Valeur de la zone *Reason* dans la sortie de l'exit
- Valeur de la zone *DataLength* dans la sortie de l'exit
- Contenu de la mémoire tampon de sortie de l'exit *OutBuffer*. Le nombre d'octets renvoyés correspond au paramètre **OutBufferLength** de l'exit le moins élevé et à la valeur de la zone *DataLength* dans la sortie de l'exit.

Si les zones *Encoding* et *CodedCharSetId* du paramètre de descripteur de message de l'exit sont les deux inchangées, le gestionnaire de files d'attente renvoie:

- Valeur des zones *Encoding* et *CodedCharSetId* dans la structure MQDXP sur l'entrée de l'exit.

Si l'une ou les deux zones *Encoding* et *CodedCharSetId* du paramètre de descripteur de message de l'exit ont été modifiées, le gestionnaire de files d'attente renvoie:

- Valeur des zones *Encoding* et *CodedCharSetId* dans le paramètre de descripteur de message de l'exit sur la sortie de l'exit

### **Echec de MQXDR\_CONVERSION\_FAILED**

La conversion a échoué.

Si l'exit spécifie cette valeur, le gestionnaire de files d'attente renvoie la valeur suivante à l'application qui a émis l'appel MQGET:

- Valeur de la zone *CompCode* dans la sortie de l'exit
- Valeur de la zone *Reason* dans la sortie de l'exit
- Valeur de la zone *DataLength* dans l'entrée de l'exit
- Contenu de la mémoire tampon d'entrée de l'exit *InBuffer*. Le nombre d'octets renvoyés est indiqué par le paramètre **InBufferLength**

Si l'exit a modifié *InBuffer*, les résultats ne sont pas définis.

*ExitResponse* est une zone de sortie de l'exit.

### **Hconn**

Type: MQHCONN

Il s'agit d'un descripteur de connexion qui peut être utilisé sur l'appel MQXCNVC. Ce descripteur n'est pas nécessairement le même que celui spécifié par l'application qui a émis l'appel MQGET.

### **pEntryPoints**

Type: PMQIEP

Adresse d'une structure MQIEP via laquelle les appels MQI et DCI peuvent être effectués.

### **Reason**

Type: MQLONG

Code anomalie qualifiant *CompCode*.

Lorsque l'exit est appelé, il contient le code anomalie renvoyé à l'application qui a émis l'appel MQGET, si l'exit choisit de ne rien faire. Parmi les valeurs possibles figurent MQRC\_TRUNCATED\_MSG\_ACCEPTED, qui indique que le message a été tronqué pour tenir dans la mémoire tampon fournie par l'application, et MQRC\_NOT\_CONVERTIS, qui indique que le message requiert une conversion mais que cette opération n'a pas encore été effectuée.

Dans la sortie de l'exit, cette zone contient la raison du renvoi à l'application dans le paramètre **Reason** de l'appel MQGET ; les éléments suivants sont recommandés:

- Si *Reason* a la valeur MQRC\_TRUNCATED\_MSG\_ACCEPTED en entrée de l'exit, les zones *Reason* et *CompCode* ne doivent pas être modifiées, que la conversion aboutisse ou échoue.

(Si la zone *CompCode* n'est pas MQCC\_OK, l'application qui extrait le message peut identifier un échec de conversion en comparant les valeurs *Encoding* et *CodedCharSetId* renvoyées dans le descripteur de message avec les valeurs demandées ; en revanche, l'application ne peut pas distinguer un message tronqué d'un message correspondant à la mémoire tampon. Pour cette raison, MQRC\_TRUNCATED\_MSG\_ACCEPTED doit être renvoyé de préférence à l'une des raisons qui indiquent l'échec de la conversion.)

- Si *Reason* avait une autre valeur en entrée de l'exit:
  - Si la conversion aboutit, *CompCode* doit être défini sur MQCC\_OK et *Reason* sur MQRC\_NONE.
  - Si la conversion échoue, ou si le message se développe et doit être tronqué pour tenir dans la mémoire tampon, *CompCode* doit être défini sur MQCC\_WARNING (ou laissé inchangé) et *Reason* sur l'une des valeurs répertoriées, pour indiquer la nature de l'échec.

Notez que si le message après la conversion est trop volumineux pour la mémoire tampon, il doit être tronqué uniquement si l'application qui a émis l'appel MQGET a spécifié l'option MQGMO\_ACCEPT\_TRUNCATED\_MSG:

- S'il a spécifié cette option, la raison MQRC\_TRUNCATED\_MSG\_ACCEPTED est renvoyée.
- S'il n'a pas spécifié cette option, le message est renvoyé non converti, avec le code anomalie MQRC\_CONVERTED\_MSG\_TOO\_BIG.

Les codes anomalie répertoriés sont recommandés par l'exit pour indiquer la raison pour laquelle la conversion a échoué, mais l'exit peut renvoyer d'autres valeurs à partir de l'ensemble de codes MQRC\_\* si cela est jugé approprié. En outre, la plage de valeurs MQRC\_APPL\_FIRST à MQRC\_APPL\_LAST est allouée pour être utilisée par l'exit pour indiquer les conditions que l'exit souhaite communiquer à l'application émettrice de l'appel MQGET.

**Remarque :** Si le message ne peut pas être converti correctement, l'exit doit renvoyer MQXDR\_CONVERSION\_FAILED dans la zone *ExitResponse* afin que le gestionnaire de files d'attente renvoie le message non converti. Cela est vrai quel que soit le code anomalie renvoyé dans la zone *Reason*.

#### **MQRC\_APPL\_FIRST**

(900, X'384') Valeur la plus faible pour le code anomalie défini par l'application.

#### **MQRC\_APPL\_LAST**

(999, X'3E7') Valeur la plus élevée pour le code anomalie défini par l'application.

#### **MQRC\_CONVERTED\_MSG\_TOO\_BIG**

(2120, X'848') Données converties trop volumineuses pour la mémoire tampon.

#### **MQRC\_NOT\_CONVERTED**

(2119, X'847') Données de message non converties.

#### **MQRC\_SOURCE\_CCSID\_ERROR**

(2111, X'83F') Identificateur de jeu de caractères codés source non valide.

#### **MQRC\_SOURCE\_DECIMAL\_ENC\_ERROR**

(2113, X'841') Codage décimal condensé dans le message non reconnu.

#### **MQRC\_SOURCE\_FLOAT\_ENC\_ERROR**

(2114, X'842') Codage à virgule flottante dans le message non reconnu.

#### **MQRC\_SOURCE\_INTEGER\_ENC\_ERROR**

(2112, X'840') Codage d'entier source non reconnu.

#### **MQRC\_TARGET\_CCSID\_ERROR**

(2115, X'843') Identificateur de jeu de caractères codés cible non valide.

#### **MQRC\_TARGET\_DECIMAL\_ENC\_ERROR**

(2117, X'845') Codage décimal condensé spécifié par le récepteur non reconnu.

#### **MQRC\_TARGET\_FLOAT\_ENC\_ERROR**

(2118, X'846') Codage à virgule flottante spécifié par le récepteur non reconnu.

#### **MQRC\_TARGET\_INTEGER\_ENC\_ERROR**

(2116, X'844') Codage d'entier cible non reconnu.

## **MQRC\_TRUNCATED\_MSG\_ACCEPTED**

(2079, X'81F') Message tronqué renvoyé (traitement terminé).

Il s'agit d'une zone d'entrée-sortie pour l'exit.

### **StrucId**

Type: MQCHAR4

Identificateur de structure. La valeur doit être:

### **ID\_STRUC\_MQDXP**

Identificateur de la structure des paramètres d'exit de conversion de données.

Pour le langage de programmation C, la constante MQDXP\_STRUC\_ID\_ARRAY est également définie ; elle a la même valeur que MQDXP\_STRUC\_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

Il s'agit d'une zone d'entrée de l'exit.

### **Version**

Type: MQLONG

Numéro de version de la structure. La valeur doit être:

### **MQDXP\_VERSION\_1**

Numéro de version de la structure des paramètres d'exit de conversion de données.

La constante suivante indique le numéro de version de la version en cours:

### **MQDXP\_CURRENT\_VERSION**

Version actuelle de la structure des paramètres d'exit de conversion de données.

**Remarque :** Lorsqu'une nouvelle version de cette structure est introduite, la présentation de la partie existante n'est pas modifiée. L'exit doit donc vérifier que la zone *Version* est égale ou supérieure à la version la plus basse qui contient les zones que l'exit doit utiliser.

Il s'agit d'une zone d'entrée de l'exit.

## **Déclaration C**

```
typedef struct tagMQDXP MQDXP;
struct tagMQDXP {
    MQCHAR4  StrucId;           /* Structure identifier */
    MQLONG   Version;          /* Structure version number */
    MQLONG   ExitOptions;      /* Reserved */
    MQLONG   AppOptions;       /* Application options */
    MQLONG   Encoding;         /* Numeric encoding required by
                               application */
    MQLONG   CodedCharSetId;   /* Character set required by application */
    MQLONG   DataLength;       /* Length in bytes of message data */
    MQLONG   CompCode;         /* Completion code */
    MQLONG   Reason;           /* Reason code qualifying CompCode */
    MQLONG   ExitResponse;     /* Response from exit */
    MQHCONN  Hconn;           /* Connection handle */
    PMQIEP   pEntryPoints;     /* Address of the MQIEP structure */
};
```

## **Déclaration COBOL ( IBM i uniquement)**

```
**  MQDXP structure
10  MQDXP.
**  Structure identifier
15  MQDXP-STRUCID      PIC X(4).
**  Structure version number
15  MQDXP-VERSION     PIC S9(9) BINARY.
**  Reserved
15  MQDXP-EXITOPTIONS PIC S9(9) BINARY.
**  Application options
15  MQDXP-APPOPTIONS PIC S9(9) BINARY.
```

```

**      Numeric encoding required by application
15 MQDXP-ENCODING      PIC S9(9) BINARY.
**      Character set required by application
15 MQDXP-CODEDCHARSETID PIC S9(9) BINARY.
**      Length in bytes of message data
15 MQDXP-DATALLENGTH  PIC S9(9) BINARY.
**      Completion code
15 MQDXP-COMPCODE     PIC S9(9) BINARY.
**      Reason code qualifying COMPCODE
15 MQDXP-REASON       PIC S9(9) BINARY.
**      Response from exit
15 MQDXP-EXITRESPONSE PIC S9(9) BINARY.
**      Connection handle
15 MQDXP-HCONN        PIC S9(9) BINARY.

```

## Déclaration de l'assembleur System/390

```

MQDXP          DSECT
MQDXP_STRUCID  DS    CL4  Structure identifier
MQDXP_VERSION  DS    F    Structure version number
MQDXP_EXITOPTIONS DS    F    Reserved
MQDXP_APOPTIONS DS    F    Application options
MQDXP_ENCODING DS    F    Numeric encoding required by application
MQDXP_CODEDCHARSETID DS    F    Character set required by application
MQDXP_DATALLENGTH DS    F    Length in bytes of message data
MQDXP_COMPCODE DS    F    Completion code
MQDXP_REASON   DS    F    Reason code qualifying COMPCODE
MQDXP_EXITRESPONSE DS    F    Response from exit
MQDXP_HCONN    DS    F    Connection handle
*
MQDXP_LENGTH   EQU    *-MQDXP
               ORG    MQDXP
MQDXP_AREA     DS    CL(MQDXP_LENGTH)

```

## MQXCNCV-Conversion de caractères

L'appel MQXCNCV convertit les caractères d'un jeu de caractères à un autre à l'aide du langage de programmation C.

Cet appel fait partie de l'interface de conversion de données IBM MQ (DCI), qui est l'une des interfaces de l'infrastructure IBM MQ .

Remarque: L'appel peut être utilisé à partir des environnements d'exit d'application et de conversion de données.

## Syntaxe

MQXCNCV (*Hconn*, *Options*, *SourceCCSID*, *SourceLength*, *SourceBuffer*, *TargetCCSID*, *TargetLength*, *TargetBuffer*, *DataLength*, *CompCode*, *Cause*)


## Paramètres

### Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente.

Dans un exit de conversion de données, Hconn est normalement le descripteur qui est transmis à l'exit de conversion de données dans la zone Hconn de la structure MQDXP ; ce descripteur n'est pas nécessairement le descripteur spécifié par l'application qui a émis l'appel MQGET.

 Sous IBM i, la valeur spéciale suivante peut être spécifiée pour Hconn:

### MQHC\_DEF\_HCONN

Descripteur de connexion par défaut.

Si vous exécutez une application CICS TS 3.2 ou supérieure, vérifiez que le programme d'exit de conversion de caractères, qui appelle l'appel MQXCNCV, est défini en tant qu'OPENAPI. Cette définition empêche l'erreur 2018 MQRC\_HCONN\_ERROR causée par une connexion incorrecte et permet à MQGET de s'exécuter.

## Options

Type : MQLONG - entrée

Options qui contrôlent l'action de MQXCNCV.

Aucune ou plusieurs des options décrites peuvent être spécifiées. Pour spécifier plusieurs options, ajoutez les valeurs ensemble (n'ajoutez pas la même constante plusieurs fois) ou combinez les valeurs à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations bit).

**Option de conversion par défaut:** L'option suivante contrôle l'utilisation de la conversion de caractères par défaut:

### MQDCC\_CONVERSION\_PAR\_DÉFAUT\_PAR\_DÉFAUT

Conversion par défaut.

Cette option indique que la conversion de caractères par défaut peut être utilisée si l'un des jeux de caractères indiqués dans l'appel, ou les deux, ne sont pas pris en charge. Cela permet au gestionnaire de files d'attente d'utiliser un jeu de caractères par défaut spécifié par l'installation qui se rapproche du jeu de caractères spécifié lors de la conversion de la chaîne.

**Remarque :** Le résultat de l'utilisation d'un jeu de caractères approximatif pour convertir la chaîne est que certains caractères peuvent être convertis de manière incorrecte. Cela peut être évité en utilisant dans la chaîne uniquement des caractères qui sont communs au jeu de caractères spécifié et au jeu de caractères par défaut.

Les jeux de caractères par défaut sont définis par une option de configuration lorsque le gestionnaire de files d'attente est installé ou redémarré.

Si MQDCC\_DEFAULT\_CONVERSION n'est pas spécifié, le gestionnaire de files d'attente utilise uniquement les jeux de caractères spécifiés pour convertir la chaîne, et l'appel échoue si l'un ou les deux jeux de caractères ne sont pas pris en charge.

Cette option est prise en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows

**Option de remplissage:** L'option suivante permet au gestionnaire de files d'attente de remplir la chaîne convertie avec des blancs ou de supprimer les caractères de fin non significatifs afin que la chaîne convertie corresponde à la mémoire tampon cible:

### MQDCC\_FILL\_TARGET\_BUFFER

Remplissage de la mémoire tampon cible.

Cette option demande que la conversion soit effectuée de telle sorte que la mémoire tampon cible soit complètement remplie:

- Si la chaîne se contracte lors de sa conversion, des blancs de fin sont ajoutés afin de remplir la mémoire tampon cible.
- Si la chaîne se développe lors de sa conversion, les caractères de fin qui ne sont pas significatifs sont supprimés pour que la chaîne convertie corresponde à la mémoire tampon cible. Si cette opération peut aboutir, l'appel se termine avec MQCC\_OK et le code anomalie MQRC\_NONE.

S'il y a trop peu de caractères de fin insignifiants, la chaîne la plus adaptée est placée dans la mémoire tampon cible et l'appel se termine avec MQCC\_WARNING et le code anomalie MQRC\_CONVERTED\_MSG\_TOO\_BIG.

Les caractères non significatifs sont les suivants:

- Blancs de fin
- Caractères suivant le premier caractère null dans la chaîne (à l'exclusion du premier caractère null lui-même)
- Si la chaîne, TargetCCSID et TargetLength sont telles que la mémoire tampon cible ne peut pas être complètement définie avec des caractères valides, l'appel échoue avec MQCC\_FAILED et le code anomalie MQRC\_TARGET\_LENGTH\_ERROR. Cela peut se produire lorsque TargetCCSID est un jeu de caractères DBCS pur (tel que UTF-16), mais que TargetLength spécifie une longueur qui est un nombre impair d'octets.
- TargetLength peut être inférieur ou supérieur à SourceLength. Au retour de MQXCNCV, DataLength a la même valeur que TargetLength.

Si cette option n'est pas spécifiée:

- La chaîne est autorisée à se contracter ou à se développer dans la mémoire tampon cible selon les besoins. Les caractères de fin non significatifs ne sont pas ajoutés ou supprimés.

Si la chaîne convertie correspond à la mémoire tampon cible, l'appel se termine avec MQCC\_OK et le code anomalie MQRC\_NONE.

Si la chaîne convertie est trop grande pour la mémoire tampon cible, une partie de la chaîne est placée dans la mémoire tampon cible et l'appel se termine avec MQCC\_WARNING et le code anomalie MQRC\_CONVERTED\_MSG\_TOO\_BIG. Notez qu'un nombre d'octets inférieur à TargetLength peut être renvoyé dans ce cas.

- TargetLength peut être inférieur ou supérieur à SourceLength. Au retour de MQXCNCV, DataLength est inférieur ou égal à TargetLength.

Cette option est prise en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows

**Options de codage:** Les options décrites peuvent être utilisées pour spécifier les codages d'entier des chaînes source et cible. Le codage approprié est utilisé uniquement lorsque l'identificateur de jeu de caractères correspondant indique que la représentation du jeu de caractères dans la mémoire principale dépend du codage utilisé pour les entiers binaires. Cela n'affecte que certains jeux de caractères multi-octets (par exemple, UTF-16).

Le codage est ignoré si le jeu de caractères est un jeu de caractères mono-octet (SBCS) ou un jeu de caractères multi-octets avec une représentation dans la mémoire principale qui ne dépend pas du codage d'entier.

Une seule des valeurs MQDCC\_SOURCE\_\* doit être spécifiée, combinée à l'une des valeurs MQDCC\_TARGET\_\*:

**MQDCC\_SOURCE\_ENC\_NATIVE**

Le codage source est la valeur par défaut pour l'environnement et le langage de programmation.

**MQDCC\_SOURCE\_ENC\_NORMAL**

Le codage source est normal.

**MQDCC\_SOURCE\_ENC\_INVERSE**

Le codage source est inversé.

**MQDCC\_SOURCE\_ENC\_UNDEFINED**

Le codage source n'est pas défini.

**MQDCC\_TARGET\_ENC\_NATIVE**

Le codage cible est la valeur par défaut pour l'environnement et le langage de programmation.

**MQDCC\_CIBLE\_ENC\_NORMAL**

Le codage cible est normal.

**MQDCC\_CIBLE\_INVERSÉ**

Le codage cible est inversé.

**MQDCC\_TARGET\_ENC\_UNDEFINED**

Le codage cible n'est pas défini.

Les valeurs de codage définies précédemment peuvent être ajoutées directement à la zone Options . Toutefois, si le codage source ou cible est obtenu à partir de la zone Encoding dans le MQMD ou d'une autre structure, le traitement suivant doit être effectué:

1. Le codage d'entier doit être extrait de la zone Encoding en éliminant les codages à virgule flottante et décimal condensé ; pour plus de détails sur cette opération, voir [«Analyse des codages»](#), à la page 946 .
2. Le codage d'entier résultant de l'étape 1 doit être multiplié par le facteur approprié avant d'être ajouté à la zone Options . Ces facteurs sont les suivants:
  - MQDCC\_SOURCE\_ENC\_FACTOR pour le codage source
  - MQDCC\_TARGET\_ENC\_FACTOR pour le codage cible

L'exemple de code suivant illustre comment cela peut être codé dans le langage de programmation C:

```
Options = (MsgDesc.Encoding & MQENC_INTEGER_MASK)
          * MQDCC_SOURCE_ENC_FACTOR
          + (DataConvExitParms.Encoding & MQENC_INTEGER_MASK)
          * MQDCC_TARGET_ENC_FACTOR;
```

S'il n'est pas spécifié, les options de codage sont définies par défaut comme non définies (MQDCC\_\*\_ENC\_UNDEFINED). Dans la plupart des cas, cela n'affecte pas la réussite de l'appel MQXCNCV. Toutefois, si le jeu de caractères correspondant est un jeu de caractères multi-octets dont la représentation dépend du codage (par exemple, un jeu de caractères UTF-16 ), l'appel échoue avec le code anomalie MQRC\_SOURCE\_INTEGER\_ENC\_ERROR ou MQRC\_TARGET\_INTEGER\_ENC\_ERROR, selon le cas.

Les options de codage sont prises en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows
-  z/OS

**Option par défaut:** si aucune des options décrites précédemment n'est spécifiée, l'option suivante peut être utilisée:

**MQDCC\_AUCUN**

Aucune option spécifiée.

MQDCC\_NONE est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette option soit utilisée avec une autre option, mais comme sa valeur est zéro, une telle utilisation ne peut pas être détectée.

**SourceCCSID**

Type : MQLONG - entrée

Il s'agit de l'identificateur de jeu de caractères codés de la chaîne d'entrée dans `SourceBuffer`.

### **SourceLength**

Type : MQLONG - entrée

Il s'agit de la longueur en octets de la chaîne d'entrée dans `SourceBuffer` ; il doit être supérieur ou égal à zéro.

### **SourceBuffer**

Type: MQCHAR x `SourceLength` -entrée

Il s'agit de la mémoire tampon contenant la chaîne à convertir d'un jeu de caractères à un autre.

### **TargetCCSID**

Type : MQLONG - entrée

Il s'agit de l'identificateur de jeu de caractères codés du jeu de caractères dans lequel `SourceBuffer` doit être converti.

### **TargetLength**

Type : MQLONG - entrée

Il s'agit de la longueur en octets de la mémoire tampon de sortie `TargetBuffer` ; il doit être supérieur ou égal à zéro. Il peut être inférieur ou supérieur à `SourceLength`.

### **TargetBuffer**

Type: MQCHAR x `TargetLength` -sortie

Il s'agit de la chaîne après sa conversion au jeu de caractères défini par `TargetCCSID`. La chaîne convertie peut être plus courte ou plus longue que la chaîne non convertie. Le paramètre **DataLength** indique le nombre d'octets valides renvoyés.

### **DataLength**

Type : MQLONG - sortie

Il s'agit de la longueur de la chaîne renvoyée dans la mémoire tampon de sortie `TargetBuffer`. La chaîne convertie peut être plus courte ou plus longue que la chaîne non convertie.

### **CompCode**

Type : MQLONG - sortie

Il peut s'agir de :

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_WARNING**

Avertissement (achèvement partiel).

#### **MQCC\_FAILED**

Echec de l'appel.

### **Motif**

Type : MQLONG - sortie

Code anomalie qualifiant `CompCode`.

Si `CompCode` est `MQCC_OK`:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si `CompCode` est `MQCC_WARNING`:

#### **MQRC\_CONVERTED\_MSG\_TOO\_BIG**

(2120, X'848') Données converties trop volumineuses pour la mémoire tampon.

Si `CompCode` est `MQCC_FAILED`:



**MQRC\_DATA\_LENGTH\_ERROR**

(2010, X'7DA') Paramètre de longueur des données non valide.

**MQRC\_DBCS\_ERROR**

(2150, X'866') Chaîne DBCS non valide.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') Descripteur de connexion non valide.

**MQRC\_OPTIONS\_ERROR**

(2046, X'7FE') Options incorrectes ou incohérentes.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') Ressources système disponibles insuffisantes.

**MQRC\_SOURCE\_BUFFER\_ERROR**

(2145, X'861') Paramètre de tampon source non valide.

**MQRC\_SOURCE\_CCSDID\_ERROR**

(2111, X'83F') Identificateur de jeu de caractères codés source non valide.

**MQRC\_SOURCE\_INTEGER\_ENC\_ERROR**

(2112, X'840') Codage d'entier source non reconnu.

**MQRC\_SOURCE\_LENGTH\_ERROR**

(2143, X'85F') Paramètre de longueur source non valide.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') Mémoire disponible insuffisante.

**MQRC\_TARGET\_BUFFER\_ERROR**

(2146, X'862') Paramètre de tampon cible non valide.

**MQRC\_TARGET\_CCSDID\_ERROR**

(2115, X'843') Identificateur de jeu de caractères codés cible non valide.

**MQRC\_TARGET\_INTEGER\_ENC\_ERROR**

(2116, X'844') Codage d'entier cible non reconnu.

**MQRC\_TARGET\_LENGTH\_ERROR**

(2144, X'860') Paramètre de longueur cible incorrect.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir [Messages et codes anomalie](#).

## Appel C

```

MQXCNV (Hconn, Options, SourceCCSID, SourceLength, SourceBuffer,
        TargetCCSID, TargetLength, TargetBuffer, &DataLength,
        &CompCode, &Reason);

```

Déclarez les paramètres comme suit :

```

MQHCONN  Hconn;           /* Connection handle */
MQLONG   Options;        /* Options that control the action of
MQXCNV */
MQLONG   SourceCCSID;    /* Coded character set identifier of string
before conversion */
MQLONG   SourceLength;   /* Length of string before conversion */
MQCHAR   SourceBuffer[n]; /* String to be converted */
MQLONG   TargetCCSID;    /* Coded character set identifier of string
after conversion */
MQLONG   TargetLength;   /* Length of output buffer */
MQCHAR   TargetBuffer[n]; /* String after conversion */
MQLONG   DataLength;     /* Length of output string */
MQLONG   CompCode;      /* Completion code */
MQLONG   Reason;        /* Reason code qualifying CompCode */

```

## Déclaration COBOL ( IBM i uniquement)

IBM i

```
CALL 'MQXCNCV' USING HCONN, OPTIONS, SOURCECCSID, SOURCELENGTH,
                    SOURCEBUFFER, TARGETCCSID, TARGETLENGTH,
                    TARGETBUFFER, DATALENGTH, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** Connection handle
01 HCONN          PIC S9(9) BINARY.
** Options that control the action of MQXCNCV
01 OPTIONS        PIC S9(9) BINARY.
** Coded character set identifier of string before conversion
01 SOURCECCSID    PIC S9(9) BINARY.
** Length of string before conversion
01 SOURCELENGTH   PIC S9(9) BINARY.
** String to be converted
01 SOURCEBUFFER    PIC X(n).
** Coded character set identifier of string after conversion
01 TARGETCCSID    PIC S9(9) BINARY.
** Length of output buffer
01 TARGETLENGTH   PIC S9(9) BINARY.
** String after conversion
01 TARGETBUFFER    PIC X(n).
** Length of output string
01 DATALENGTH    PIC S9(9) BINARY.
** Completion code
01 COMPCODE        PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON          PIC S9(9) BINARY.
```

## Déclaration d'assembleur S/390

```
CALL MQXCNCV, (HCONN, OPTIONS, SOURCECCSID, SOURCELENGTH,          X
              SOURCEBUFFER, TARGETCCSID, TARGETLENGTH, TARGETBUFFER, X
              DATALENGTH, COMPCODE, REASON)
```

Déclarez les paramètres comme suit :

HCONN	DS	F	Connection handle
OPTIONS	DS	F	Options that control the action of MQXCNCV
SOURCECCSID	DS	F	Coded character set identifier of string before conversion
*			
SOURCELENGTH	DS	F	Length of string before conversion
SOURCEBUFFER	DS	CL(n)	String to be converted
TARGETCCSID	DS	F	Coded character set identifier of string after conversion
*			
TARGETLENGTH	DS	F	Length of output buffer
TARGETBUFFER	DS	CL(n)	String after conversion
DATALENGTH	DS	F	Length of output string
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## MQ\_DATA\_CONV\_EXIT-Exit de conversion de données

L'appel MQ\_DATA\_CONV\_EXIT décrit les paramètres transmis à l'exit de conversion de données.

Aucun point d'entrée appelé MQ\_DATA\_CONV\_EXIT n'est fourni par le gestionnaire de files d'attente (voir la remarque d'utilisation [11](#)).

Cette définition fait partie de l'interface de conversion de données IBM MQ (DCI), qui est l'une des interfaces de l'infrastructure IBM MQ.

## Syntaxe

MQ\_DATA\_CONV\_EXIT (*DataConvExitParms*, *MsgDesc*, *InBufferLength*, *InBuffer*, *OutBufferLength*, *OutBuffer*)

## Paramètres

### DataConvExitParms

Type: MQDXP-entrée / sortie

Cette structure contient des informations relatives à l'appel de l'exit. L'exit définit des informations dans cette structure pour indiquer le résultat de la conversion. Pour plus de détails sur les zones de cette structure, voir «MQDXP-Paramètre d'exit de conversion de données», à la page 958 .

### MsgDesc

Type : MQMD - entrée/sortie

En entrée de l'exit, il s'agit du descripteur de message associé aux données de message transmises à l'exit dans le paramètre **InBuffer** .

**Remarque :** Le paramètre **MsgDesc** transmis à l'exit est toujours la version la plus récente de MQMD prise en charge par le gestionnaire de files d'attente qui appelle l'exit. Si l'exit est destiné à être portable entre différents environnements, l'exit vérifie la zone *Version* dans *MsgDesc* pour vérifier que les zones auxquelles l'exit doit accéder sont présentes dans la structure.

Dans les environnements suivants, l'exit est transmis à un MQMD version-2 :

-  AIX
-  IBM i
-  Linux
-  Windows

Dans tous les autres environnements qui prennent en charge l'exit de conversion de données, l'exit est transmis à un MQMD version-1 .

En sortie, l'exit remplace les zones *Encoding* et *CodedCharSetId* par les valeurs demandées par l'application, si la conversion a abouti ; ces modifications sont répercutées sur l'application. Toutes les autres modifications apportées par l'exit à la structure sont ignorées ; elles ne sont pas reflétées dans l'application.

Si l'exit renvoie MQXDR\_OK dans la zone *ExitResponse* de la structure MQDXP, mais qu'il ne modifie pas les zones *Encoding* ou *CodedCharSetId* du descripteur de message, le gestionnaire de files d'attente renvoie pour ces zones les valeurs que les zones correspondantes de la structure MQDXP avaient en entrée de l'exit.

### InBuffer

Type : MQLONG - entrée

Longueur en octets de *InBuffer*.

Il s'agit de la longueur de la mémoire tampon d'entrée *InBuffer*, qui indique le nombre d'octets à traiter par l'exit. *InBufferLength* est la longueur la moins élevée des données de message avant la conversion et la longueur de la mémoire tampon fournie par l'application lors de l'appel MQGET.

La valeur est toujours supérieure à zéro.

### InBuffer

Type: MQBYTEInBufferLength -entrée

Mémoire tampon contenant le message non converti.

Contient les données du message avant la conversion. Si l'exit ne parvient pas à convertir les données, le gestionnaire de files d'attente renvoie le contenu de cette mémoire tampon à l'application une fois l'exit terminé.

**Remarque :** L'exit ne doit pas modifier `InBuffer` ; si ce paramètre est modifié, les résultats ne sont pas définis.

Dans le langage de programmation C, ce paramètre est défini comme un pointeur vers vide.

### **OutBuffer-Longueur**

Type : MQLONG - entrée

Longueur en octets de `OutBuffer`.

Il s'agit de la longueur de la mémoire tampon de sortie `OutBuffer`, qui est identique à la longueur de la mémoire tampon fournie par l'application sur l'appel `MQGET`.

La valeur est toujours supérieure à zéro.

### **OutBuffer**

Type: MQBYTExOutBufferLength -sortie

Mémoire tampon contenant le message converti.

En sortie de l'exit, si la conversion a abouti (comme indiqué par la valeur `MQXDR_OK` dans la zone `ExitResponse` du paramètre **DataConvExitParms** ), `OutBuffer` contient les données de message à distribuer à l'application, dans la représentation demandée. Si la conversion a échoué, les modifications apportées par l'exit à cette mémoire tampon sont ignorées.

Dans le langage de programmation C, ce paramètre est défini comme un pointeur vers vide.

## **Remarques sur l'utilisation**

1. Un exit de conversion de données est un exit écrit par l'utilisateur qui reçoit le contrôle lors du traitement d'un appel `MQGET`. La fonction exécutée par l'exit de conversion de données est définie par le fournisseur de l'exit ; cependant, l'exit doit se conformer aux règles décrites ici et dans la structure de paramètres associée `MQDXP`.

Les langages de programmation pouvant être utilisés pour un exit de conversion de données sont déterminés par l'environnement.

2. L'exit est appelé uniquement si toutes les instructions suivantes sont vraies:

- L'option `MQGMO_CONVERT` est spécifiée dans l'appel `MQGET`
- La zone `Format` du descripteur de message n'est pas `MQFMT_NONE`
- Le message n'est pas déjà dans la représentation requise, c'est-à-dire que l'un ou les deux éléments `CodedCharSetId` et `Encoding` du message sont différents de la valeur spécifiée par l'application dans le descripteur de message fourni dans l'appel `MQGET`.
- Le gestionnaire de files d'attente n'a pas déjà effectué la conversion avec succès
- La longueur de la mémoire tampon de l'application est supérieure à zéro
- La longueur des données de message est supérieure à zéro
- Le code anomalie jusqu'à présent lors de l'opération `MQGET` est `MQRC_NONE` ou `MQRC_TRUNCATED_MSG_ACCEPTED`

3. Lorsqu'un exit est en cours d'écriture, pensez à le coder de manière à ce qu'il puisse convertir les messages qui ont été tronqués. Les messages tronqués peuvent se produire de l'une des manières suivantes:

- L'application de réception fournit une mémoire tampon plus petite que le message, mais spécifie l'option `MQGMO_ACCEPT_TRUNCATED_MSG` dans l'appel `MQGET`.

Dans ce cas, la zone `Reason` du paramètre **DataConvExitParms** en entrée de l'exit a la valeur `MQRC_TRUNCATED_MSG_ACCEPTED`.

- L'expéditeur du message l'a tronqué avant de l'envoyer. Cela peut se produire avec des messages de rapport, par exemple (voir «[Conversion des messages de rapport](#)», à la page 957 pour plus de détails).

Dans ce cas, la zone Reason du paramètre **DataConvExitParms** en entrée de l'exit a la valeur MQRC\_NONE (si l'application de réception a fourni une mémoire tampon suffisamment grande pour le message).

Par conséquent, la valeur de la zone Reason en entrée de l'exit ne peut pas toujours être utilisée pour déterminer si le message a été tronqué.

La caractéristique distinctive d'un message tronqué est que la longueur fournie à l'exit dans le paramètre **InBufferLength** est inférieure à la longueur impliquée par le nom de format contenu dans la zone Format du descripteur de message. L'exit doit donc vérifier la valeur de InBufferLength avant de tenter de convertir les données ; il ne doit pas supposer que la quantité complète de données impliquée par le nom de format a été fournie.

Si l'exit n'a pas été écrit pour convertir les messages tronqués et que InBufferLength est inférieur à la valeur attendue, l'exit renvoie MQXDR\_CONVERSION\_FAILED dans la zone ExitResponse du paramètre **DataConvExitParms** , avec les zones CompCode et Reason définies sur MQCC\_WARNING et MQRC\_FORMAT\_ERROR.

Si l'exit a été écrit pour convertir des messages tronqués, il convertit autant de données que possible (voir la note d'utilisation suivante), en prenant soin de ne pas tenter d'examiner ou de convertir des données au-delà de la fin de InBuffer. Si la conversion aboutit, l'exit laisse la zone Reason dans le paramètre **DataConvExitParms** inchangée. Renvoie MQRC\_TRUNCATED\_MSG\_ACCEPTED si le message a été tronqué par le gestionnaire de files d'attente du récepteur et MQRC\_NONE si le message a été tronqué par l'expéditeur du message.

Il est également possible qu'un message se développe lors de la conversion, au point où il est supérieur à OutBuffer. Dans ce cas, l'exit doit décider s'il faut tronquer le message ; la zone AppOptions dans le paramètre **DataConvExitParms** indique si l'application de réception a spécifié l'option MQGMO\_ACCEPT\_TRUNCATED\_MSG.

4. En règle générale, toutes les données du message fourni à l'exit dans InBuffer sont converties, ou aucune de ces données n'est convertie. Cependant, une exception à cette règle se produit si le message est tronqué, soit avant la conversion, soit pendant la conversion ; dans ce cas, il peut y avoir un élément incomplet à la fin de la mémoire tampon (par exemple: 1 octet d'un caractère codé sur deux octets, ou 3 octets d'un entier codé sur quatre octets). Dans ce cas, envisagez d'omettre l'élément incomplet et de définir les octets inutilisés dans OutBuffer sur des valeurs nulles. Toutefois, les éléments ou caractères complets d'un tableau ou d'une chaîne doivent être convertis.
5. Lorsqu'un exit est nécessaire pour la première fois, le gestionnaire de files d'attente tente de charger un objet portant le même nom que le format (à l'exception des extensions). L'objet chargé doit contenir l'exit qui traite les messages avec ce nom de format. Pensez à rendre le nom de l'exit et le nom de l'objet contenant l'exit identiques, bien que tous les environnements n'en aient pas besoin.
6. Une nouvelle copie de l'exit est chargée lorsqu'une application tente d'extraire le premier message qui utilise cette Format depuis que l'application s'est connectée au gestionnaire de files d'attente. Pour les applications CICS ou IMS , cela signifie que le sous-système CICS ou IMS est connecté au gestionnaire de files d'attente. Une nouvelle copie peut également être chargée à d'autres moments, si le gestionnaire de files d'attente a supprimé une copie précédemment chargée. Pour cette raison, un exit ne doit pas tenter d'utiliser la mémoire statique pour communiquer des informations d'un appel de l'exit au suivant-l'exit peut être déchargé entre les deux appels.
7. S'il existe un exit fourni par l'utilisateur portant le même nom que l'un des formats intégrés pris en charge par le gestionnaire de files d'attente, l'exit fourni par l'utilisateur ne remplace pas la routine de conversion intégrée. Les seules circonstances dans lesquelles un tel exit est appelé sont les suivantes:
  - Si la routine de conversion intégrée ne peut pas gérer les conversions vers ou depuis le CodedCharSetId ou Encoding impliqué, ou


- Si la routine de conversion intégrée n'a pas réussi à convertir les données (par exemple, parce qu'une zone ou un caractère ne peut pas être converti).
8. La portée de l'exit dépend de l'environnement. Les noms Format doivent être choisis pour réduire le risque de conflits avec d'autres formats. Commencez par les caractères qui identifient l'application définissant le nom de format.
  9. L'exit de conversion de données s'exécute dans un environnement tel que celui du programme qui a émis l'appel MQGET ; l'environnement inclut l'espace adresse et le profil utilisateur (le cas échéant). Le programme peut être un agent de canal de transmission de messages envoyant des messages à un gestionnaire de files d'attente de destination qui ne prend pas en charge la conversion de messages. L'exit ne peut pas compromettre l'intégrité du gestionnaire de files d'attente, car il ne s'exécute pas dans l'environnement du gestionnaire de files d'attente.
  10. Le seul appel MQI pouvant être utilisé par l'exit est MQXCNCV ; la tentative d'utilisation d'autres appels MQI échoue avec le code anomalie MQRC\_CALL\_IN\_PROGRESS ou d'autres erreurs imprévisibles.
  11. Aucun point d'entrée appelé MQ\_DATA\_CONV\_EXIT n'est fourni par le gestionnaire de files d'attente. Toutefois, un typedef est fourni pour le nom MQ\_DATA\_CONV\_EXIT dans le langage de programmation C, qui peut être utilisé pour déclarer l'exit écrit par l'utilisateur afin de s'assurer que les paramètres sont corrects. Le nom de l'exit doit être identique au nom de format (nom contenu dans la zone Format de MQMD), bien que cela ne soit pas obligatoire dans tous les environnements.

L'exemple suivant montre comment l'exit qui traite le format MYFORMAT peut être déclaré dans le langage de programmation C:

```
#include "cmqc.h"
#include "cmqxc.h"

MQ_DATA_CONV_EXIT MYFORMAT;

void MQENTRY MYFORMAT(
    PMQDXP  pDataConvExitParms, /* Data-conversion exit parameter
                                block */
    PMQMD   pMsgDesc,           /* Message descriptor */
    MQLONG  InBufferLength,     /* Length in bytes of InBuffer */
    PMQVOID pInBuffer,         /* Buffer containing the unconverted
                                message */
    MQLONG  OutBufferLength,    /* Length in bytes of OutBuffer */
    PMQVOID pOutBuffer)        /* Buffer containing the converted
                                message */
{
    /* C language statements to convert message */
}
```

12.  Sous z/OS, si un exit de croisement d'API est également en vigueur, il est appelé après l'exit de conversion de données.

## Appel C

```
exitname (&DataConvExitParms, &MsgDesc, InBufferLength,
          InBuffer, OutBufferLength, OutBuffer);
```

Les paramètres transmis à l'exit sont déclarés comme suit:

```
MQDXP  DataConvExitParms; /* Data-conversion exit parameter block */
MQMD   MsgDesc;           /* Message descriptor */
MQLONG InBufferLength;    /* Length in bytes of InBuffer */
MQBYTE InBuffer[n];      /* Buffer containing the unconverted
                           message */
MQLONG OutBufferLength;  /* Length in bytes of OutBuffer */
MQBYTE OutBuffer[n];     /* Buffer containing the converted
                           message */
```

## Déclaration COBOL (IBM i uniquement)

IBM i

```
CALL 'exitname' USING DATACONVEXITPARMS, MSGDESC, INBUFFERLENGTH,  
INBUFFER, OUTBUFFERLENGTH, OUTBUFFER.
```

Les paramètres transmis à l'exit sont déclarés comme suit:

```
** Data-conversion exit parameter block  
01 DATACONVEXITPARMS.  
COPY CMQDXPV.  
** Message descriptor  
01 MSGDESC.  
COPY CMQMDV.  
** Length in bytes of INBUFFER  
01 INBUFFERLENGTH PIC S9(9) BINARY.  
** Buffer containing the unconverted message  
01 INBUFFER PIC X(n).  
** Length in bytes of OUTBUFFER  
01 OUTBUFFERLENGTH PIC S9(9) BINARY.  
** Buffer containing the converted message  
01 OUTBUFFER PIC X(n).
```

## Déclaration de l'assembleur System/390

```
CALL EXITNAME, (DATACONVEXITPARMS, MSGDESC, INBUFFERLENGTH, X  
INBUFFER, OUTBUFFERLENGTH, OUTBUFFER)
```

Les paramètres transmis à l'exit sont déclarés comme suit:

```
DATACONVEXITPARMS CMQDXPA , Data-conversion exit parameter block  
MSGDESC CMQMDA , Message descriptor  
INBUFFERLENGTH DS F Length in bytes of INBUFFER  
INBUFFER DS CL(n) Buffer containing the unconverted  
* message  
OUTBUFFERLENGTH DS F Length in bytes of OUTBUFFER  
OUTBUFFER DS CL(n) Buffer containing the converted  
* message
```

## Propriétés spécifiées en tant qu'éléments MQRFH2

Les propriétés de descripteur de non-message peuvent être spécifiées en tant qu'éléments dans les dossiers d'en-tête MQRFH2 . Présentation des éléments MQRFH2 spécifiés en tant que propriétés.

Cela permet de conserver la compatibilité avec les versions précédentes des clients IBM MQ JMS et XMS . Cette section explique comment spécifier des propriétés dans les en-têtes MQRFH2 .

Pour utiliser les éléments MQRFH2 comme propriétés, spécifiez les éléments comme décrit dans [Utilisation de IBM MQ classes for Java](#) . Ces informations complètent celles décrites dans «[MQRFH2 - En-tête 2 de règles et de formatage](#)», à la page 552.

## Mappage de types de données de propriété à des types de données MQRFH2

Cette rubrique fournit des informations sur les types de propriété de message mappés à leurs types de données MQRFH2 correspondants.

Type de propriété de message	Type de données MQRFH2
MQBYTE [ ]	bin.hex
MQBOOL	boolean

Tableau 635. Types de données MQRFH2 pris en charge (suite)

Type de propriété de message	Type de données MQRFH2
MQINT8	i1
MQINT16	i2
MQINT32	i4
MQINT64	i8
MQFLOAT32	r4
MQFLOAT64	r8
MQCHAR [ ]	chaîne

Tout élément sans type de données est supposé être de type "chaîne".

Un type de données MQRFH2 de `int`, c'est-à-dire un entier de taille non spécifiée, est traité comme s'il s'agissait d'un `i8`.

Une valeur nulle est indiquée par l'attribut d'élément `xsi:nil='true'`. N'utilisez pas l'attribut `xsi:nil='false'` pour les valeurs non nulles.

Par exemple, la propriété suivante a une valeur null:

```
<NullProperty xsi:nil='true'></NullProperty>
```

Une propriété d'octet ou de chaîne de caractères peut avoir une valeur vide. Il est représenté par un élément MQRFH2 avec une valeur d'élément de longueur nulle.

Par exemple, la propriété suivante a une valeur vide:

```
<EmptyProperty></EmptyProperty>
```

## Dossiers MQRFH2 pris en charge

Présentation de l'utilisation des zones de descripteur de message en tant que propriétés.

Les dossiers `<jms>`, `<mcd>`, `<mqext>` et `<usr>` sont décrits dans [L'en-tête MQRFH2 et JMS](#). Le dossier `<usr>` est utilisé pour transporter toutes les propriétés définies par l'application JMS qui sont associées à un message. Les groupes ne sont pas autorisés dans le dossier `<usr>`.

L'en-tête MQRFH2 et JMS prennent en charge les dossiers supplémentaires suivants:

- `<mq>`

Ce dossier est utilisé et réservé aux propriétés définies par MQ qui sont utilisées par IBM MQ.

- `<mq_usr>`

Ce dossier peut être utilisé pour transporter des propriétés définies par l'application qui ne sont pas exposées en tant que propriétés définies par l'utilisateur JMS, car les propriétés peuvent ne pas répondre aux exigences d'une propriété JMS. Ce dossier peut contenir des groupes que le dossier `<usr>` ne peut pas contenir.

- Tout dossier marqué avec l'attribut `content='properties'`.

Ce type de dossier est équivalent au dossier `<mq_usr>` dans le contenu.

- `<mmps>`

Ce dossier est utilisé pour les propriétés de publication / abonnement IBM MQ.

IBM MQ prend également en charge les dossiers suivants qui sont déjà utilisés par WAS/SIB:



- <sib>

Ce dossier est utilisé et réservé aux propriétés de message système WAS/SIB qui ne sont pas exposées en tant que propriétés JMS ou qui sont mappées aux propriétés JMS\_IBM\_\*, mais qui sont exposées aux applications WAS/SIB ; ces dernières incluent les propriétés des chemins de routage de réacheminement et de réacheminement inverse.

Au moins certains ne peuvent pas être exposés en tant que propriétés JMS, car il s'agit de tableaux d'octets. Si votre application ajoute des propriétés à ce dossier, la valeur est ignorée ou supprimée.

- <sib\_usr>

Ce dossier est utilisé et réservé aux propriétés de message utilisateur WAS/SIB qui ne peuvent pas être exposées en tant que propriétés utilisateur JMS car elles ne sont pas de types pris en charge ; elles sont exposées aux applications WAS/SIB.

Il s'agit de propriétés utilisateur que vous pouvez obtenir ou définir via l'interface SIMessage, mais le contenu du tableau d'octets est mappé à la valeur de propriété requise.

Si votre application IBM MQ écrit un élément bin.hex arbitraire dans le dossier, l'application reçoit probablement un IOException, car il n'est pas au format attendu pour la restauration. Si vous ajoutez autre chose qu'un élément bin.hex, vous recevez un ClassCastException.

N'essayez pas de rendre les propriétés disponibles pour WAS/SIB en utilisant ce dossier ; utilisez plutôt le dossier <usr> à cette fin.

- <sib\_context>

Ce dossier est utilisé pour les propriétés de message système WAS/SIB qui ne sont pas exposées aux applications utilisateur WAS/SIB ou en tant que propriétés JMS. Il s'agit notamment des propriétés de sécurité et transactionnelles utilisées pour les services Web et similaires.

Votre application ne doit pas ajouter de propriétés à ce dossier.

- <mqema>

Ce dossier a été utilisé par WAS/SIB à la place du dossier <mqext> .

Les noms de dossier MQRFH2 sont sensibles à la casse.

Les dossiers suivants sont réservés, dans n'importe quel mélange de minuscules ou de majuscules:

- Tout dossier préfixé par mq ou wmq ; réservé à l'utilisation par IBM MQ.
- Tout dossier préfixé par sib ; réservé à l'utilisation par WAS/SIB.
- Dossiers <Root> et <Body> ; réservés mais non utilisés.

Les dossiers suivants ne sont pas reconnus comme contenant des propriétés de message:

- <psc>

Utilisé par IBM Integration Bus pour transmettre des messages de commande de publication / abonnement au courtier.

- <pscr>

Utilisé par IBM Integration Bus pour contenir des informations provenant du courtier, en réponse à des messages de commande de publication / abonnement.

- Tout dossier non défini par IBM, qui n'est pas marqué avec l'attribut content='properties' .

Ne spécifiez pas content='properties' dans les dossiers <psc> ou <pscr> . Dans ce cas, ces dossiers sont traités comme des propriétés et IBM Integration Bus risque de ne plus fonctionner comme prévu.

Si votre application génère des messages avec des propriétés, dans les en-têtes MQRFH2 à reconnaître comme un en-tête MQRFH2 contenant des propriétés, l'en-tête doit figurer dans la liste des en-têtes pouvant être chaînés en tête du message.

MQRFH2 peut être précédé d'un nombre quelconque d'en-têtes standard MQH, d'un MQCIH, d'un MQDLH, d'un MQIIH, d'un MQTM, d'un MQTMC2 ou d'un MQXQH. Une chaîne ou un MQCFH se termine l'analyse syntaxique car ils ne peuvent pas être chaînés.

Il est possible qu'un message contienne plusieurs en-têtes MQRFH2 contenant toutes des propriétés de message. Les dossiers portant le même nom peuvent coexister dans des en-têtes différents, sauf restriction contraire, par exemple par WAS/SIB. Les dossiers sont traités comme un seul dossier logique, s'ils sont tous dans des en-têtes significatifs.

Alors que les dossiers des en-têtes significatifs ne peuvent pas être fusionnés avec ceux des en-têtes non significatifs, les dossiers portant le même nom dans les en-têtes significatifs peuvent être fusionnés, ce qui supprime les propriétés en conflit. Vos applications ne doivent pas dépendre de la présentation des propriétés dans leur message.

Les groupes MQRFH2 sont analysés pour rechercher des propriétés dans les dossiers définis par l'utilisateur, c'est-à-dire dans les dossiers <wmq>, <jms>, <mcd>, <usr>, <mqext>, <sib>, <sib\_usr>, <sib\_context> et <mqema>.

Les groupes des dossiers de propriétés définis par IBM, à l'exception des dossiers <wmq> et <mq>, sont analysés pour les propriétés.

Un dossier MQRFH2 ne peut pas contenir de contenu mixte ; un dossier ou un groupe peut contenir des groupes ou des propriétés, ou une valeur, mais pas les deux.

Un segment d'un message, qu'il s'agisse du premier segment ou d'un segment suivant, ne peut pas contenir de propriétés définies par IBM MQ autres que celles du descripteur de message. Par conséquent, l'insertion d'un message contenant de telles propriétés avec MQMF\_SEGMENT ou MQMF\_SEGMENTATION\_ALLOWED entraîne l'échec de l'insertion avec MQRC\_SEGMENTATION\_NOT\_ALLOWED.


Toutefois, les groupes de messages peuvent contenir des propriétés définies par IBM MQ.

## Génération d'en-têtes MQRFH2

Si IBM MQ convertit les propriétés de message en leur représentation MQRFH2, il doit ajouter MQRFH2 au message. Il ajoute le fichier MQRFH2 sous la forme d'un en-tête distinct ou le fusionne avec un en-tête existant.

La génération de nouveaux en-têtes MQRFH2 par IBM MQ peut perturber les en-têtes existants dans un message. Les applications qui analysent une mémoire tampon de messages pour les en-têtes doivent savoir que le nombre et la position des en-têtes dans une mémoire tampon peuvent changer dans certaines circonstances. IBM MQ tente de réduire l'impact de l'ajout de propriétés à un message en fusionnant les propriétés de message dans un en-tête MQRFH2 existant, là où il le peut. Il tente également de minimiser l'impact en insérant un MQRFH2 généré dans une position fixe par rapport à d'autres en-têtes dans la mémoire tampon du message.

Un en-tête MQRFH2 généré est placé après le MQMD, ainsi que n'importe quel nombre d'en-têtes MQXQH, MQRFH et MQDLH, quel que soit l'ordre dans lequel ils se trouvent. L'en-tête MQRFH2 généré est placé immédiatement avant le premier en-tête qui n'est pas un en-tête MQMD, MQXQH, MQDLH ou MQRFH.

 Sur les systèmes z/OS, l'en-tête MQRFH2 généré est créé dans le fichier CCSID de l'application. Ceci est défini comme suit:

- Pour les applications par lots LE utilisant l'interface DLL, CCSID est le CODESET associé à l'environnement local en cours lors de l'émission de **MQCONN** (la valeur par défaut est 1047).
- Pour les applications par lots LE liées à l'un des stubs MQ par lots, CCSID est le paramètre CODESET associé à l'environnement local en cours au moment du premier appel MQI émis après **MQCONN** (la valeur par défaut est 1047).
- Pour les applications par lots non LE s'exécutant sur une unité d'exécution z/OS UNIX System Services (z/OS UNIX), CCSID est la valeur de THLICCSID au moment du premier appel MQI émis après **MQCONN** (la valeur par défaut est 1047).
- Pour les autres applications par lots, CCSID est le CCSID du gestionnaire de files d'attente.

Pour les applications LE, l'environnement local peut être modifié à l'aide du service d'appels `setlocale()` / `CEESETL LE`. Pour les applications non LE s'exécutant sur des unités d'exécution z/OS UNIX, la valeur de `THLICCSID` peut être modifiée à l'aide de la z/OS UNIX macro de mappage **BPXYTHLI**.

## Règles de fusion des MQRFH2 générés

Les règles suivantes s'appliquent à la fusion d'un MQRFH2 généré avec un MQRFH2 existant. L'en-tête MQRFH2 généré est fusionné avec un en-tête MQRFH2 existant, si:

1. Le MQRFH2 existant se trouve à la même position que IBM MQ place un MQRFH2 généré ou plus tôt dans la chaîne d'en-tête.
2. Le CCSID des propriétés générées est identique au `CCSIDNameValue` du MQRFH2 existant.

Dans le cas contraire, l'en-tête généré est placé séparément dans la mémoire tampon, dans la position décrite précédemment.

## Règles de fusion des dossiers dans un MQRFH2 existant

Si les propriétés de message sont fusionnées dans un MQRFH2 existant, le MQRFH2 existant est analysé pour rechercher les dossiers qui correspondent aux propriétés de message et les fusionne. Si un dossier correspondant n'existe pas, un nouveau dossier est ajouté à la fin des dossiers existants. Si un dossier correspondant existe, la recherche est effectuée dans le dossier. Toutes les propriétés correspondantes sont écrasées. Tous les nouveaux sont ajoutés à la fin du dossier.

## Restrictions du dossier MQRFH2

Présentation des restrictions de dossier dans les en-têtes MQRFH2

Les restrictions MQRFH2 s'appliquent aux dossiers suivants:

- Les noms d'élément du dossier `<us1>` ne doivent pas commencer par le préfixe JMS ; ces noms de propriété sont réservés à l'utilisation par JMS et ne sont pas valides pour les propriétés définies par l'utilisateur.

Un tel nom d'élément n'entraîne pas l'échec de l'analyse syntaxique de MQRFH2, mais il n'est pas accessible aux API de propriété de message IBM MQ.

- Les noms d'élément du dossier `<us1>` ne doivent pas être, dans une combinaison de minuscules ou de majuscules, NULL, TRUE, FALSE, NOT, AND, OR, BETWEEN, LIKE, IN, IS et ESCAPE. Ces noms correspondent à des mots clés SQL et rendent les sélecteurs d'analyse syntaxique plus difficiles, car `<us1>` est le dossier par défaut utilisé lorsqu'aucun dossier n'est spécifié pour une propriété particulière dans un sélecteur.

Un tel nom d'élément n'entraîne pas l'échec de l'analyse syntaxique de MQRFH2, mais il n'est pas accessible aux API de propriété de message IBM MQ.

- Le modèle de contenu du dossier `<us1>` est le suivant:
  - Tout nom XML valide peut être utilisé comme nom d'élément, à condition qu'il ne contienne pas de signe deux-points.
  - Seuls les éléments simples, et non les dossiers imbriqués, sont autorisés.
  - Tous les éléments prennent le type de chaîne par défaut, sauf s'ils sont modifiés par un attribut `dt="xxx"`.
  - Tous les éléments sont facultatifs, mais ne doivent pas apparaître plus d'une fois dans un dossier.
- Les noms d'élément d'un dossier considéré comme contenant des propriétés de message ne doivent pas contenir de point (.) (caractère Unicode U+002E), car il est utilisé dans les noms de propriété pour indiquer la hiérarchie.

Un tel nom d'élément n'entraîne pas l'échec de l'analyse syntaxique de MQRFH2, mais il n'est pas accessible aux API de propriété de message IBM MQ.

En général, les en-têtes MQRFH2 qui contiennent des données de style XML valides peuvent être analysés par IBM MQ sans échec, bien que certains éléments de MQRFH2 ne soient pas accessibles via les API de propriété de message IBM MQ .

## Conflits de noms d'élément MQRFH2

Présentation des conflits dans les noms d'élément MQRFH2 .

Une seule valeur peut être associée à une propriété de message. Si une tentative d'accès à une propriété entraîne un conflit de valeurs, l'une d'elles est choisie de préférence à l'autre.

La syntaxe IBM MQ permettant d'accéder aux éléments MQRFH2 permet l'identification unique d'un élément, si un dossier ne contient aucun élément portant le même nom. Si un dossier contient plusieurs éléments portant le même nom, la valeur de la propriété utilisée est celle la plus proche de la tête du message.

Cela s'applique si plusieurs dossiers du même nom sont contenus dans des en-têtes MQRFH2 significatifs différents dans le même message.

Un conflit peut se produire lorsque l'appel MQGET est traité après qu'une propriété de descripteur de non-message a été définie deux fois: via un appel MQSETMP et directement dans l'en-tête MQRFH2 brut.

Si cela se produit, la propriété associée au message par un appel API prend la préférence sur une dans les données de message, c'est-à-dire sur celle de l'en-tête MQRFH2 brut. Si un conflit se produit, il est considéré comme venant logiquement avant les données du message.

## Mappage des noms de propriété vers les noms de dossier et d'élément MQRFH2

Présentation des différences entre les noms de propriété et les noms d'élément dans l'en-tête MQRFH2 .

Lorsque vous utilisez l'une des API définies qui génèrent des en-têtes MQRFH2 , afin de spécifier des propriétés de message (par exemple, MQ JMS), le nom de propriété n'est pas nécessairement le nom d'élément dans le dossier MQRFH2 .

Par conséquent, un mappage est effectué entre le nom de la propriété et l'élément MQRFH2 , et inversement, en prenant en compte à la fois le nom du dossier qui contient l'élément et le nom de l'élément. Certains exemples de IBM MQ classes for JMS sont déjà documentés dans [Utilisation de IBM MQ classes for Java](#).

Nom de la propriété	Nom de dossier MQRFH2	Nom d'élément MQRFH2
JMSDestination	jms	Dst
JMSType	mcd	Type, Set, Fmt
xxx (défini par l'utilisateur, où xxx ne commence pas par JMS)	usr	xxx

Par conséquent, lorsqu'une application JMS accède à la propriété JMSDestination , elle est mappée à l'élément Dst dans le dossier <jms> .

Lorsque vous spécifiez des propriétés en tant qu'éléments MQRFH2 , IBM MQ définit ses éléments comme suit:

Nom de la propriété	Nom de dossier MQRFH2	Nom de groupe MQRFH2	Nom d'élément MQRFH2
<Property>	<usr>	Non applicable	<Property>
<folder>.<Property>	<folder>	Non applicable	<Property>
<folder>.<group>.<Property>	<folder>	<group>	<Property>

Par exemple, lorsqu'une application IBM MQ tente d'accéder à la propriété Property1 , elle est mappée à l'élément Property1 dans le dossier <usr> . La propriété wmq .Property2 est mappée à la propriété Property2 dans le dossier <wmq> .

Si le nom de la propriété en contient plusieurs , le nom d'élément MQRFH2 utilisé est celui qui suit la valeur finale. et les groupes MQRFH2 sont utilisés pour former une hiérarchie ; les groupes MQRFH2 imbriqués sont autorisés.

L'en-tête JMS et les propriétés spécifiques au fournisseur contenues dans MQRFH2 dans les dossiers <mcd>, <jms>et <mqext> sont accessibles par une application IBM MQ à l'aide des noms abrégés définis dans Utilisation de IBM MQ classes for Java .

Les propriétés JMS définies par l'utilisateur sont accessibles à partir du dossier <usr> . Une application IBM MQ peut utiliser le dossier <usr> pour ses propriétés d'application s'il est acceptable que la propriété apparaisse aux applications JMS comme l'une de ses propriétés définies par l'utilisateur.

Si ce n'est pas acceptable, choisissez un autre dossier ; le dossier <wmq\_usr> est fourni comme emplacement standard pour ces propriétés nonJMS .

Vos applications peuvent spécifier et utiliser n'importe quel dossier MQRFH2 avec une utilisation bien définie, non documentée dans «Propriétés spécifiées en tant qu'éléments MQRFH2», à la page 975 si vous notez ce qui suit:

1. Il se peut que le dossier soit déjà utilisé ou qu'il soit utilisé ultérieurement par une autre application fournissant un accès non défini aux propriétés qu'il contient. Voir Noms de propriété pour connaître la convention de dénomination suggérée pour les noms de propriété.
2. Les propriétés ne sont pas accessibles aux versions précédentes du client IBM MQ classes for JMS ou XMS qui peuvent uniquement accéder au dossier <usr> pour les propriétés définies par l'utilisateur
3. Le dossier doit être marqué avec l'attribut content avec la valeur définie sur properties, par exemple, content='properties' .

«MQSETMP-Définition de la propriété de message», à la page 816 ajoute automatiquement cet attribut si nécessaire. Cet attribut ne doit être ajouté à aucun des dossiers définis par IBM, par exemple, <jms> et <usr>. Dans ce cas, le message est rejeté par le client IBM MQ classes for JMS avant IBM WebSphere MQ 7.0. avec un MessageFormatException.

Etant donné que le dossier <usr> est l'emplacement par défaut des propriétés de la syntaxe <Property> , une application IBM MQ et une application JMS doivent accéder à la même valeur de propriété définie par l'utilisateur en utilisant le même nom.

## Noms de dossier réservés

Il existe plusieurs noms de dossier réservés. Vous ne pouvez pas utiliser de tels noms comme préfixes de dossier ; par exemple, Root .Property1 n'accède pas à une propriété valide car Root est réservé. La liste suivante contient les noms de dossiers réservés:

- Racine
- Corps
- Propriétés
- Environnement
- LocalEnvironment
- DestinationList
- ExceptionList
- InputBody
- InputRoot
- InputProperties
- Environnement InputLocal
- Liste InputDestination

- Liste InputException
- OutputRoot
- Environnement OutputLocal
- Liste OutputDestination
- Liste OutputException

## Mapper des zones de descripteur de propriété dans les en-têtes MQRFH2

Lorsqu'une propriété est convertie en élément MQRFH2, les attributs d'élément suivants sont utilisés pour spécifier les zones significatives du descripteur de propriété: Cette section décrit comment les zones MQPD sont converties en attributs d'élément MQRFH2.

### Support

La zone de descripteur de propriété de support est divisée en trois attributs d'élément

- L'attribut d'élément **sr** spécifie des valeurs dans le masque de bit MQPD\_REJECT\_UNSUP\_MASK.
- L'attribut d'élément **sa** spécifie des valeurs dans le masque de bits MQPD\_ACCEPT\_UNSUP\_MASK.
- L'attribut d'élément **sx** spécifie des valeurs dans le masque de bit MQPD\_ACCEPT\_UNSUP\_IF\_XMIT\_MASK.

Ces attributs d'élément ne sont valides que dans le dossier < mq > et sont ignorés s'ils sont définis sur des éléments des autres dossiers contenant des propriétés.

*Tableau 638. Zones MQPD mappées aux attributs d'élément MQRFH2*

Valeur de prise en charge	Attribut d'élément MQRFH2	Valeur de l'attribut MQRFH2
MQPD_SUPPORT_XX_ENCODE_CASE_ONE facultatif	sa	facultatif Il s'agit de la valeur par défaut.
MQPD_SUPPORT_REQUIS	sr	obligatoire
MQPD_SUPPORT_REQUIRED_IF_LOCAL	sx	local

### Contexte

Utilisez l'attribut d'élément **context** pour indiquer le contexte de message auquel appartient une propriété. Utilisez une seule valeur. Cet attribut d'élément est valide sur une propriété dans tout dossier contenant des propriétés.

*Tableau 639. Valeurs de contexte mappées aux valeurs d'attribut MQRFH2*

Valeur du contexte	Valeur de l'attribut MQRFH2
MQPD_NO_CONTEXT	Aucun Il s'agit de la valeur par défaut.
MQPD_USER_CONTEXT	utilisateur

### CopyOptions

Utilisez l'attribut d'élément **copy** pour indiquer les messages dans lesquels une propriété doit être copiée. Plusieurs valeurs sont admises ; séparez les valeurs par une virgule. Par exemple, **copy='reply'** et **copy='publish,report'** sont valides. Cet attribut d'élément est valide sur une propriété dans tout dossier contenant des propriétés.

**Remarque :** Dans la définition d'attribut, les apostrophes ou les guillemets sont valides, par exemple **copy='reply'** ou **copy="report"**

<i>Tableau 640. Valeurs CopyOption mappées aux valeurs d'attribut MQRFH2</i>	
<b>Valeur CopyOption</b>	<b>Valeur de l'attribut MQRFH2</b>
MQPD_COPY_FORWARD	réacheminement
MQPD_RÉPONSE_COPIE	réponse
RAPPORT MQPD_COPY_REPORT	rapport
MQPD_COPY_PUBLISH	publier
MQPD_COPY_ALL	tout Ne spécifiez pas cette valeur avec une autre valeur. Lorsqu'elle est utilisée avec une autre valeur, elle est prioritaire sur toute valeur à l'exception de <b>none</b> .
MQPD_COPY_DEFAULT	par défaut Il s'agit de la valeur par défaut. Cela revient à spécifier les trois valeurs MQCOPY_FORWARD, MQCOPY_REPORT et MQCOPY_PUBLISH. Ne spécifiez pas cette valeur avec une autre valeur.
MQPD_COPY_NONE	Aucun Ne spécifiez pas cette valeur avec une autre valeur. Lorsqu'elle est utilisée avec une autre valeur, elle est prioritaire.

## **Restrictions du dossier < mq> MQRFH2**

Lorsqu'un message est inséré dans une file d'attente, il est recherché dans un dossier < mq> afin que le message puisse être traité en fonction de ses propriétés définies par MQ. Pour permettre une analyse syntaxique efficace des propriétés définies par MQ, les restrictions suivantes s'appliquent au dossier:

- Seules les propriétés du premier dossier < mq> significatif du message sont traitées par MQ; les propriétés de tout autre dossier < mq> du message sont ignorées.
- Si le dossier est en UTF-8, seuls les caractères UTF-8 mono-octet sont autorisés dans le dossier. Un caractère multi-octet dans le dossier peut entraîner l'échec de l'analyse syntaxique et le rejet du message.
- N'incluez pas les groupes MQRFH2 dans le dossier < mq>. La présence du caractère Unicode U+003C dans une valeur de propriété entraîne le rejet du message.
- N'utilisez pas de chaînes d'échappement dans le dossier. Une chaîne d'échappement est traitée comme la valeur réelle de l'élément.
- Seul le caractère Unicode U+0020 est traité comme un espace dans le dossier. Tous les autres caractères sont traités comme significatifs et peuvent entraîner l'échec de l'analyse syntaxique du dossier et le rejet du message.

Si l'analyse du dossier < mq> échoue ou si le dossier ne respecte pas ces restrictions, le message est rejeté avec CompCode **MQCC\_FAILED** et Reason **MQRC\_RFH\_RESTRICTED\_FORMAT\_ERR**.

## **En-têtes MQRFH2 non valides**

Lors d'un appel MQPUT, MQPUT1 ou MQGET, une analyse partielle des en-têtes MQRFH2 du message peut être effectuée pour vérifier quels dossiers sont inclus et pour déterminer si les dossiers contiennent des propriétés. Présentation des en-têtes MQRFH2 non valides.

Si l'analyse partielle du message ne peut pas aboutir car la structure n'est pas valide, par exemple, la zone StrucLength est trop petite, alors:

- L'appel MQPUT ou MQPUT1 échoue avec le code anomalie MQRC\_RFH\_ERROR, s'il est possible de déterminer que l'application inclut une option IBM WebSphere MQ 7 , de sorte que les applications existantes n'échouent pas.
- L'appel MQGET est renvoyé avec succès et l'en-tête MQRFH2 contenant l'erreur est renvoyé dans la mémoire tampon que vous avez fournie.

Si l'analyse syntaxique partielle échoue car il est impossible de détecter si un dossier particulier contient des propriétés ou non, par exemple, le dossier commence par <<jms. L'analyse syntaxique échoue donc avant que le nom du dossier ne soit déterminé.

- L'appel MQPUT ou MQPUT1 échoue avec le code anomalie MQRC\_RFH\_FORMAT\_ERROR, s'il est possible de déterminer que l'application inclut une option IBM WebSphere MQ 7 , de sorte que les applications existantes n'échouent pas.
- L'appel MQGET est renvoyé avec succès et l'en-tête MQRFH2 contenant l'erreur est renvoyé dans la mémoire tampon que vous avez fournie.
- En interne dans le gestionnaire de files d'attente, le message n'est pas rejeté en raison du dossier mal formaté, mais le dossier est toujours traité comme s'il ne contenait aucune propriété.

Un message peut circuler dans le réseau du gestionnaire de files d'attente avec un dossier contenant une telle erreur de syntaxe, mais il n'est jamais analysé et détecté, alors qu'un ou plusieurs dossiers du message sont:

- Valide
- L'analyse syntaxique a abouti
- Utilisé dans le traitement du message

Par conséquent, la détection n'est pas garantie.

Si l'une de vos applications utilise «MQSETMP-Définition de la propriété de message», à la page 816 ou MQINQMP pour accéder à une propriété, ce qui entraîne l'analyse complète d'un dossier MQRFH2 et la détection d'une erreur telle que l'analyse ne peut pas aboutir, cela est indiqué par un code retour approprié à l'appel d'API. Aucune propriété du dossier n'est mise à la disposition de l'application.

Si une tentative d'analyse complète d'un dossier MQRFH2 est effectuée et que l'analyseur syntaxique trouve des attributs d'élément non reconnus ou un type de données non reconnu, l'analyse se poursuit et aboutit sans qu'aucun avertissement ne soit émis ; cela ne constitue pas une erreur d'analyse.

## Conversion de page de code

Cette section décrit les noms de jeu de codes et les CCSID, la langue nationale, la conversion z/OS , la conversion IBM i , et la prise en charge de la conversion Unicode.


Chaque section de la langue nationale contient les informations suivantes:

- Les CCSID natifs pris en charge
- Conversions de page de codes non prises en charge

Les termes suivants sont utilisés dans les informations:

 **AIX**

Indique IBM MQ for AIX.

 **Linux**

Indique IBM MQ for Linux for Intel et IBM MQ for Linux for zSeries.

 **OS/400**

Indique IBM MQ for IBM i.



## Windows Windows

Indique IBM MQ for Windows.

## z/OS z/OS

Indique IBM MQ for z/OS.

La valeur par défaut pour la conversion de données est la conversion à effectuer sur le système cible (récepteur).

Si le produit source prend en charge la conversion, un canal peut être configuré et les données échangées en définissant l'attribut de canal CONVERT sur YES au niveau de la source.

### Remarque :

1. La conversion des informations IBM MQ MQI client s'effectue sur le serveur. Par conséquent, le serveur doit prendre en charge la conversion du CCSID client en CCSID serveur.
2. La conversion peut inclure la prise en charge ajoutée par CSD/PTF à la dernière version de IBM MQ. Vérifiez le contenu du dernier niveau de service pour savoir si vous devez installer une CSD/PTF pour activer cette conversion.
3. Le CCSID du gestionnaire de files d'attente IBM MQ doit être Mixte ou SBSCS.
4. Certains CCSID, par exemple 850 sous AIX, qui ne sont pas pris en charge par le système d'exploitation peuvent toujours être utilisés par l'application et peuvent également être définis comme CCSID du gestionnaire de files d'attente IBM MQ . Cela est autorisé uniquement à des fins de compatibilité avec les versions antérieures et la conversion échouera si les tables de conversion appropriées ne sont pas installées.

Pour obtenir une référence croisée entre certains numéros de CCSID et certains noms de jeu de codes de l'industrie, voir [Tableau 641](#), à la page 985 .

### Référence associée

«Langues nationales», à la page 986

Ces informations contiennent les langues prises en charge par IBM MQ.

## Noms de jeu de codes et CCSID

Noms de jeu de codes et CCSID correspondants pour chaque nom de jeu de codes.

z/OS IBM MQ for z/OS fournit plus de conversions que celles répertoriées dans les tables spécifiques à la langue. Pour obtenir la liste complète des conversions, voir [Tableau 674](#), à la page 1012.

Noms de jeu de codes	CCSID
ISO 8859-1	819
ISO 8859-2	912
ISO 8859-3	913
ISO 8859-5	915
ISO 8859-6	1089
ISO 8859-7	813
ISO 8859-8	916
ISO 8859-9	920
ISO 8859-13	921
ISO 8859-15 (euro)	923

Tableau 641. Noms de jeu de codes et CCSID (suite)

Noms de jeu de codes	CCSID
big5	950
eucJP	954 5050 33722
eucKR	970
eucTW	964
eucCN	1383
PCK	943
GBK	1386
koi8-r	878

## Langues nationales

Ces informations contiennent les langues prises en charge par IBM MQ.






Les langues prises en charge par IBM MQ sont les suivantes:

- Anglais (Etats-Unis)-voir la rubrique [«Anglais \(Etats-Unis\)»](#), à la page 987
- Allemand-voir la rubrique [«Allemand»](#), à la page 987
- Danois et norvégien-voir la rubrique [«Danois et norvégien»](#), à la page 988
- Finnois et suédois-voir la rubrique [«Finnois et suédois»](#), à la page 989
- Italien-voir la rubrique [«Italien»](#), à la page 990
- Espagnol-voir la rubrique [«Espagnol»](#), à la page 990
- Anglais / gaélique du Royaume-Uni-voir la rubrique [«Anglais / gaélique du Royaume-Uni»](#), à la page 991
- Français-voir la rubrique [«Français»](#), à la page 992
- Multilingue-voir la rubrique [«Multilingues»](#), à la page 992
- Portugais-voir la rubrique [«Portugais»](#), à la page 993
- Islandais-voir la rubrique [«Islandais»](#), à la page 994
- Langues d'Europe orientale-voir rubrique [«Langues d'Europe orientale»](#), à la page 994
- Cyrillique-voir la rubrique [«alphabet cyrillique»](#), à la page 996
- Estonien-voir la rubrique [«Estonien»](#), à la page 996
- Letton et lituanien-voir la rubrique [«Letton et lituanien»](#), à la page 997
- Ukrainien-voir la rubrique [«Ukrainien»](#), à la page 999
- Grec-voir la rubrique [«Grec»](#), à la page 999
- Turc-voir la rubrique [«Turc»](#), à la page 1000
- Hébreu-voir la rubrique [«Hébreu»](#), à la page 1000
- Farsi-voir la rubrique [«Persan»](#), à la page 1002
- Urdu-voir la rubrique [«Ourdou»](#), à la page 1003
- Thaï-voir la rubrique [«Thaïlandais»](#), à la page 1003
- Lao-voir la rubrique [«Laotien»](#), à la page 1004
- Vietnamien-voir la rubrique [«Vietnamien»](#), à la page 1004
- Japonais latin SBCS-voir la rubrique [«Japonais latin SBCS»](#), à la page 1005
- Japonais Katakana SBCS-voir la rubrique [«Japonais Katakana SBCS»](#), à la page 1006

- Japonais Kanji / latin Mixte-voir la rubrique «Japonais Kanji / Latin Mixte», à la page 1007
- Japonais Kanji / Katakana Mixte-voir la rubrique «Japonais Kanji / Katakana Mixte», à la page 1008
- Coréen-voir la rubrique «Coréen», à la page 1009
- Chinois simplifié-voir la rubrique «Chinois simplifié», à la page 1010
- Chinois traditionnel-voir la rubrique «chinois traditionnel», à la page 1011

## Anglais (Etats-Unis)

Détails des CCSID et de la conversion de CCSID pour l'anglais américain.

<i>Tableau 642. CCSID natifs pour l'anglais américain sur les plateformes prises en charge</i>	
Plateforme	CCSID natifs
 IBM i  z/OS	37, 924, 1140
 AIX  Windows	819, 923, 5348 437, 850, 1252, 5348, 858
 Linux	819, 923
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

### IBM i



Page de codes :

#### 37

Ne convertit pas en pages de codes 923, 858

#### 924

Ne convertit pas en pages de codes 437, 858, 1051, 1140, 1252, 1275, 5348

#### 1140

Ne convertit pas en pages de codes 924, 1051, 1275

### Allemand

Détails des CCSID et de la conversion de CCSID pour l'allemand.






<i>Tableau 643. CCSID natifs pour l'allemand sur les plateformes prises en charge</i>	
Plateforme	CCSID natifs
 IBM i  z/OS	273, 924, 1141
 AIX  Windows	819, 923, 5348 437, 850, 858, 1252, 5348

Tableau 643. CCSID natifs pour l'allemand sur les plateformes prises en charge (suite)

Plateforme	CCSID natifs
 Linux	819, 923
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## IBM i



Page de codes :

### 273

Ne convertit pas en pages de codes 858, 923, 924, 1275

### 924

Ne convertit pas en pages de codes 273, 437, 858, 1051, 1141, 1252, 1275, 5348






### 1141

Ne convertit pas en pages de codes 924, 1051, 1275

## Danois et norvégien

Détails des CCSID et de la conversion de CCSID pour le danois et le norvégien.

Tableau 644. CCSID natifs pour le danois et le norvégien sur les plateformes prises en charge

Plateforme	CCSID natifs
 IBM i	277, 924, 1142
 z/OS	
 AIX	819, 923, 5348
 Windows	850, 858, 865, 1252, 5348
 Linux	819, 923
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## IBM i



Page de codes :

### 277

Ne convertit pas en pages de codes 858, 923, 924, 1275

### 924

Ne convertit pas en pages de codes 277, 858, 865, 1051, 1142, 1252, 1275, 5348

### 1142

Ne convertit pas en pages de codes 924, 865, 1051, 1275

## AIX



Page de codes :

### 819

Ne convertit pas en page de codes 865

## Windows



Page de codes :

### 865

Ne convertit pas en pages de codes 1051, 1275

## Finnois et suédois

Détails des CCSID et de la conversion de CCSID pour le finnois et le suédois.

Tableau 645. CCSID natifs pour le finnois et le suédois sur les plateformes prises en charge

Plateforme	CCSID natifs
IBM i z/OS	278, 924, 1143
AIX	819, 923, 5348
Windows	437, 850, 858, 865, 1252, 5348
Linux	819, 923
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## IBM i



Page de codes :

### 278

Ne convertit pas en pages de codes 858, 923, 924, 1275

### 924

Ne convertit pas en pages de codes 278, 437, 858, 865, 1051, 1143, 1252, 1275, 5348

### 1143

Ne convertit pas en pages de codes 865, 924, 1051, 1275

## AIX



Page de codes :

### 819

Ne convertit pas en page de codes 865

## 850

Ne convertit pas en page de codes 865

## Windows



Page de codes :

## 865

Ne convertit pas en pages de codes 1051, 1275

## Italien

Détails des CCSID et de la conversion de CCSID pour l'italien.

Tableau 646. CCSID natifs pour l'italien sur les plateformes prises en charge

Plateforme	CCSID natifs
IBM i z/OS	280, 924, 1144
AIX	819, 923, 5348
Windows	437, 850, 858, 1252, 5348
Linux	819, 923
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## IBM i



Page de codes :

## 280

Ne convertit pas en pages de codes 858, 923, 924, 1275

## 924

Ne convertit pas en pages de codes 280, 437, 858, 1051, 1144, 1252, 1275, 5348

## 1144

Ne convertit pas en pages de codes 924, 1051, 1275



## Espagnol

Détails des CCSID et de la conversion de CCSID pour l'espagnol.

Tableau 647. CCSID natifs pour l'espagnol sur les plateformes prises en charge

Plateforme	CCSID natifs
IBM i z/OS	284, 924, 1145
AIX	819, 923, 5348

Tableau 647. CCSID natifs pour l'espagnol sur les plateformes prises en charge (suite)

Plateforme	CCSID natifs
 Windows	437, 850, 858, 1252, 5348
 Linux	819, 923
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## IBM i



Page de codes :

### 284

Ne convertit pas en pages de codes 858, 923, 924, 1275

### 924

Ne convertit pas en pages de codes 284, 437, 858, 1051, 1145, 1252, 1275, 5348






### 1145

Ne convertit pas en pages de codes 924, 1051, 1275

## Anglais / gaélique du Royaume-Uni

Détails des CCSID et de la conversion de CCSID pour l'anglais / gaélique britannique.

Tableau 648. CCSID natifs pour l'anglais/gaélique britannique sur les plateformes prises en charge

Plateforme	CCSID natifs
 IBM i	285, 924, 1146
 z/OS	
 AIX	819, 923, 5348
 Windows	437, 850, 858, 1252, 5348
 Linux	819, 923
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## IBM i



Page de codes :

### 285

Ne convertit pas en pages de codes 858, 923, 924, 1275

### 924






Ne convertit pas en pages de codes 285, 437, 858, 1051, 1146, 1252, 1275, 5348

## 1146

Ne convertit pas en pages de codes 924, 1051, 1275

### Français

Détails des CCSID et de la conversion de CCSID pour le français.

Plateforme	CCSID natifs
 IBM i  z/OS	297, 924, 1147
 AIX	819, 923, 5348
 Windows	437, 850, 858, 1252, 5348
 Linux	819, 923
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

### IBM i



Page de codes :

#### 297

Ne convertit pas en pages de codes 858, 923, 924, 1275, 5348

#### 924






Ne convertit pas en pages de codes 297, 437, 858, 1051, 1147, 1252, 1275, 5348

#### 1147

Ne convertit pas en pages de codes 924, 1051, 1275

### Multilingues

Détails des CCSID et de la conversion de CCSID pour Multilingue.

Plateforme	CCSID natifs
 IBM i  z/OS	500, 924, 1148
 AIX	819, 923, 5348
 Windows	437, 850, 858, 1252, 5348
 Linux	819, 923
Client Apple	1275



Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## IBM i



Page de codes :

### 500

Ne convertit pas en pages de codes 858, 923

### 924

Ne convertit pas en pages de codes 437, 858, 1051, 1148, 1252, 1275, 5348

### 1148

Ne convertit pas en pages de codes 924, 1051, 1275

## Portugais

Détails des CCSID et de la conversion de CCSID pour le portugais.

<i>Tableau 651. CCSID natifs pour le portugais sur les plateformes prises en charge</i>	
Plateforme	CCSID natifs
IBM i	37, 500, 924, 1140
IBM i	500, 924, 1140
AIX	819, 923, 5348
Windows	850, 858, 860, 1252, 5348
Linux	819, 923
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## IBM i



Page de codes :

### 37

Ne convertit pas en pages de codes 858, 923, 1275

### 500

Ne convertit pas en pages de codes 858, 923, 1275

### 924

Ne convertit pas en pages de codes 858, 860, 1051, 1140, 1252, 1275, 5348

### 1140

Ne convertit pas en pages de codes 860, 924, 1051, 1275

## Windows



Page de codes :






## 860

Ne convertit pas en pages de codes 1051, 1275

### Islandais

Détails des CCSID et de la conversion de CCSID pour l'islandais.

Tableau 652. CCSID natifs pour l'islandais sur les plateformes prises en charge

Plateforme	CCSID natifs
 IBM i  z/OS	871, 924, 1149
 AIX	819, 923, 5348
 Windows	850, 858, 861, 1252, 5348
 Linux	819, 923
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

### IBM i



Page de codes :

#### 871

Ne convertit pas en pages de codes 858, 923, 924, 1275, 5348

#### 924

Ne convertit pas en pages de codes 858, 861, 871, 1051, 1149, 1252, 1275, 5348

#### 1149

Ne convertit pas en pages de codes 924, 1051, 1275

### Windows



Page de codes :

#### 861

Ne convertit pas en pages de codes 1051, 1275

### Langues d'Europe orientale

Détails des CCSID et de la conversion de CCSID pour les langues d'Europe de l'Est. Les langues typiques utilisant ces CCSID sont l'albanais, le croate, le tchèque, le hongrois, le polonais, le roumain, le serbe, le slovaque et le slovène.

Tableau 653. CCSID natifs pour les langues d'Europe de l'Est sur les plateformes prises en charge






Plateforme	CCSID natifs
 IBM i  z/OS	870, 1153

Tableau 653. CCSID natifs pour les langues d'Europe de l'Est sur les plateformes prises en charge (suite)

Plateforme	CCSID natifs
 Windows	852, 1250, 5346, 9044
 AIX	912
 Linux	
Client Apple d'Europe de l'Est	1282
Client Apple roumain	1285
Client Apple croate	1284

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## z/OS



Page de codes :

### 870

Ne convertit pas en pages de codes 1284, 1285

### 1153

Ne convertit pas en pages de codes 1250, 1284, 1285

## IBM i



Page de codes :

### 870

Ne convertit pas en pages de codes 1284, 1285, 5346, 9044

### 1153

Ne convertit pas en pages de codes 1282, 1284, 1285, 5346, 9044

## , Linux



Page de codes :

### 912

Ne convertit pas en pages de codes 1284, 1285

## Windows



Page de codes :

### 852

Ne convertit pas en pages de codes 1284, 1285

### 1250

Ne convertit pas en pages de codes 1284, 1285






### 9044

Ne convertit pas en pages de codes 912, 1282, 1284, 1285

## alphabet cyrillique

Détails des CCSID et de la conversion de CCSID pour Cyrillique. Les langues typiques utilisant ces CCSID sont le biélorusse, le bulgare, le macédonien, le russe et le serbe.

Tableau 654. CCSID natifs pour Cyrillique sur les plateformes prises en charge

Plateforme	CCSID natifs
 z/OS	1025
 IBM i	880, 1025
 Windows	855, 866, 1131, 1251, 5347
 AIX	915
 Linux	
Client Apple	1283

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

### IBM i



Page de codes :

#### 880

Ne convertit pas en pages de codes 855, 866, 878, 1131, 5347

#### 1025

Ne convertit pas en pages de codes 878, 5347

### Windows



Page de codes :

#### 855

Ne convertit pas en page de codes 1131

#### 866

Ne convertit pas en page de codes 1131

#### 1131

Ne convertit pas en pages de codes 855, 866, 880, 1283

### Estonien

Détails des CCSID et de la conversion de CCSID pour l'estonien.

Tableau 655. CCSID natifs pour l'estonien sur les plateformes prises en charge






Plateforme	CCSID natifs
 IBM i	1122, 1157
 z/OS	
 Windows	902, 922, 1257, 5353, 9449

Tableau 655. CCSID natifs pour l'estonien sur les plateformes prises en charge (suite)

Plateforme	CCSID natifs
 AIX	902, 922
 Linux	

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## z/OS



Page de codes :

### 1122

Ne convertit pas en pages de codes 902, 1157, 9449

### 1157

Ne convertit pas en pages de codes 922, 1122, 1257, 9449

## IBM i



Page de codes :

### 1122

Ne convertit pas en pages de codes 902, 5353, 9449

### 1157

Ne convertit pas en pages de codes 922, 5353, 9449

## Linux



Page de codes :

### 902

Ne convertit pas en pages de codes 922, 1122, 9449

### 922

Ne convertit pas en pages de codes 902, 1157, 9449

## Windows



Page de codes :

### 5353

Ne convertit pas en page de codes 9449

### 9449

Ne convertit pas en pages de codes 902, 922, 1122, 1157, 1257, 5353






### 902

Ne convertit pas en pages de codes 922, 1122, 9449

## Letton et lituanien

Détails des CCSID et de la conversion de CCSID pour le letton et le lituanien.

Tableau 656. CCSID natifs pour letton et lituanien sur les plateformes prises en charge

Plateforme	CCSID natifs
 IBM i  z/OS	1112, 1156
 Windows	901, 921, 1257, 5353, 9449
 AIX  Linux	901, 921

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

### z/OS



Page de codes :

#### 1112

Ne convertit pas en pages de codes 901, 1156, 9449

#### 1156

Ne convertit pas en pages de codes 901, 1156, 9449

### IBM i



Page de codes :

#### 1112

Ne convertit pas en page de codes 5353

#### 1153

Ne convertit pas en pages de codes 921, 5353, 9449

### Linux



Page de codes :

#### 902

Ne convertit pas en pages de codes 921, 1112, 1257, 9449

#### 921

Ne convertit pas en pages de codes 901, 1156, 9449

### Windows



Page de codes :

#### 901

Ne convertit pas en pages de codes 921, 1112, 1257, 9449

#### 5355






Ne convertit pas en page de codes 9449

## 9449

Ne convertit pas en pages de codes 901, 921, 1112, 1156, 1257

## Ukrainien

Détails des CCSID et de la conversion de CCSID pour l'ukrainien.

Plateforme	CCSID natifs
 IBM i  z/OS	1123
 Windows	1124, 1125, 1251, 5347
 AIX  Linux	1124

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## IBM i



Page de codes :

### 1123

Ne convertit pas en page de codes 5347

## Windows



Page de codes :

### 1125

Ne convertit pas en page de codes 1123

## Grec

Détails des CCSID et de la conversion de CCSID pour le grec.






Plateforme	CCSID natifs
 IBM i  z/OS	875
 Windows	869, 1253, 5349
 AIX  Linux RCN	813
Client Apple	1280

Tableau 658. CCSID natifs pour le grec sur les plateformes prises en charge (suite)

Plateforme	CCSID natifs
Client DOS	737

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs, les CCSID natifs des autres plateformes avec les exceptions suivantes.

## IBM i



Page de codes :

### 875

Ne convertit pas en page de codes 5349

## Windows



Page de codes :

### 1253

Ne convertit pas en page de codes 737

### 5349

Ne convertit pas en page de codes 737

## Turc

Détails des CCSID et de la conversion de CCSID pour le turc.

Tableau 659. CCSID natifs pour le turc sur les plateformes prises en charge

Plateforme	CCSID natifs
IBM i z/OS	1026
Windows	857, 1254, 5350
AIX Linux	920
Client Apple	1281

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## IBM i



Page de codes :

### 1026






Ne convertit pas en page de codes 5350

## Hébreu

Détails des CCSID et de la conversion de CCSID pour l'hébreu.



Tableau 660. CCSID natifs pour l'hébreu sur les plateformes prises en charge

Plateforme	CCSID natifs
 z/OS	424, 803, 4899, 12712
 IBM i	424
 AIX	916, 9048
 Windows	1255, 5351
 Linux	916

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## z/OS



Page de codes :

### 424

Ne convertit pas en pages de codes 867, 4899, 9048, 12712

### 803

Ne convertit pas en pages de codes 867, 4899, 5351, 9048, 12712

### 4899

Ne convertit pas en pages de codes 424, 803, 856, 862, 916, 1255

### 12712

Ne convertit pas en pages de codes 424, 803, 856, 916, 1255

## IBM i



Page de codes :

### 424

Ne convertit pas en pages de codes 803, 867, 4899, 5351, 9048, 12712

La page de codes 424 est également convertie vers et depuis le CCSID 4952, qui est une variante de 856.

## AIX



Page de codes :

### 916

Ne convertit pas en pages de codes 867, 4899, 9048, 12712

### 9048

Ne convertit pas en pages de codes 424, 803, 856, 862, 916, 1255

## Windows



Page de codes :

## 1255






Ne convertit pas en pages de codes 867, 4899, 9048, 12712

## 5351

Ne convertit pas en page de codes 803

## Arabe

Détails des CCSID et de la conversion de CCSID pour l'arabe

<i>Tableau 661. CCSID natifs pour l'arabe sur les plateformes prises en charge</i>	
Plateforme	CCSID natifs
 IBM i	420
 z/OS	
 AIX	1046, 1089
	1089 (voir note)
 Windows	720, 864, 1256, 5352
 Linux	1089

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## IBM i



Page de codes :

### 420

Ne convertit pas en page de codes 5352

## Linux, Tru64



Page de codes :

### 1089

Ne convertit pas en page de codes 720

## Windows



Page de codes :

### 720

Ne convertit pas en pages de codes 1089, 5352






### 5352

Ne convertit pas en page de codes 720

## Persan

Détails des CCSID et de la conversion de CCSID pour Farsi.

Tableau 662. CCSID natifs pour Farsi sur les plateformes prises en charge

Plateforme	CCSID natifs
 IBM i  z/OS	1097
 AIX  Linux  Windows	1098 (voir note)






**Remarque :** Le CCSID natif de ces plateformes n'a pas été normalisé et peut être modifié.

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes.

### Ourdou

Détails des CCSID et de la conversion de CCSID pour l'ourdou.

Tableau 663. CCSID natifs pour l'ourdou sur les plateformes prises en charge

Plateforme	CCSID natifs
 IBM i  z/OS	918
 Windows	868
 AIX  Linux	1006

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

### IBM i



Page de codes :

#### 918

Ne convertit pas en page de codes 1006

### Thaïlandais

Détails des CCSID et de la conversion de CCSID pour le thaï.

Tableau 664. CCSID natifs pour le thaï sur les plateformes prises en charge






Plateforme	CCSID natifs
 IBM i  z/OS	838

Tableau 664. CCSID natifs pour le thaï sur les plateformes prises en charge (suite)

Plateforme	CCSID natifs
 AIX  Linux  Windows	874 (voir note)






**Remarque :** Le CCSID natif de ces plateformes n'a pas été normalisé et peut être modifié.

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes.

### Laotien

Détails des CCSID et de la conversion de CCSID pour le laotien.

Tableau 665. CCSID natifs pour le laotien sur les plateformes prises en charge






Plateforme	CCSID natifs
 IBM i  z/OS	1132
 AIX  Linux  Windows	1133

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes.

### Vietnamien

Détails des CCSID et de la conversion de CCSID pour le vietnamien.

Tableau 666. CCSID natifs pour le vietnamien sur les plateformes prises en charge

Plateforme	CCSID natifs
 IBM i  z/OS	1130
 Windows	1258, 5354
 AIX  Linux	1129

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

### IBM i








Page de codes :

### 1130



Ne convertit pas en pages de codes 1129, 5354

## Japonais latin SBCS

Détails des CCSID et de la conversion de CCSID pour les SBCS latins japonais.

Plateforme	CCSID natifs
 IBM i	1027
 z/OS	
 AIX	932, 5050, 33722 (voir la note 1)
 Windows	932, 943 (voir note 2)
 Linux	943, 5050

### Remarque :

-  5050 et 33722 sont des CCSID liés à la page de codes de base 954 sur AIX. Le CCSID signalé par le système d'exploitation est 33722.
-  Windows NT utilise la page de codes 932, mais celle-ci est mieux représentée par le CCSID 943. Toutefois, toutes les plateformes de IBM MQ ne prennent pas en charge ce CCSID.

Sous IBM MQ for Windows , le CCSID 932 est utilisé pour représenter la page de codes 932, mais une modification du fichier `./conv/table/ccsid.tbl` peut être effectuée, qui modifie le CCSID utilisé en 943.

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## z/OS



Page de codes :

### 1027

Non converti en pages de codes 932, 942, 943, 954, 5050, 33722

## IBM i



Page de codes :

### 1027

Ne convertit pas en page de codes 932

## AIX



Page de codes :

### 932

Ne convertit pas en page de codes 1027

## 5050

Ne convertit pas en page de codes 1027

## 33722

Ne convertit pas en page de codes 1027

## Linux



Page de codes :

## 943

Ne convertit pas en page de codes 1027

## 5050

Ne convertit pas en page de codes 1027

## Japonais Katakana SBCS

Détails des CCSID et de la conversion de CCSID pour le japonais Katakana SBCS.

Plateforme	CCSID natifs
IBM i	290
z/OS	
AIX	932, 5050, 33722 (voir la note 1)
Windows	932, 943 (voir note 2)
Linux	943, 5050

### Remarque :

- 5050 et 33722 sont des CCSID liés à la page de codes de base 954 sur AIX. Le CCSID signalé par le système d'exploitation est 33722.
- Windows NT utilise la page de codes 932, mais celle-ci est mieux représentée par le CCSID 943. Toutefois, toutes les plateformes de IBM MQ ne prennent pas en charge ce CCSID.  
Sous IBM MQ for Windows , le CCSID 932 est utilisé pour représenter la page de codes 932, mais une modification du fichier `./conv/table/ccsid.tbl` peut être effectuée, qui modifie le CCSID utilisé en 943.
- Outre les conversions précédentes, IBM MQ prend en charge la conversion du CCSID 897 en CCSID 37, 273, 277, 278, 280, 284, 285, 290, 297, 437, 500, 819, 850, 1027 et 1252 sur les plateformes suivantes:
  - AIX
  - Linux

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## z/OS



Page de codes :

## 290

Ne convertit pas en pages de codes 932, 943, 954, 5050, 33722

## IBM i

▶ IBM i

Page de codes :

## 290

Ne convertit pas en page de codes 932

## AIX

▶ AIX

Page de codes :

## 932

Ne convertit pas en pages de codes 290, 897

## 5050

Ne convertit pas en pages de codes 290, 897

## 33722

Ne convertit pas en pages de codes 290, 897

## Linux

▶ Linux

Page de codes :

## 943

Ne convertit pas en pages de codes 290, 897

## 5050

Ne convertit pas en pages de codes 290, 897

## Japonais Kanji / Latin Mixte

Détails des CCSID et de la conversion de CCSID pour le japonais Kanji / Latin Mixed.

Plateforme	CCSID natifs
▶ IBM i IBM i ▶ z/OS z/OS	1399, 5035 (voir la note 1)
▶ AIX AIX	932, 5050, 33722 (voir la remarque 2)
▶ Windows Windows	932, 943 (voir note 4)
▶ Linux Linux	943, 5050

## Remarque :

- ▶ IBM i ▶ z/OS 5035 est un CCSID associé à la page de codes 939
- ▶ AIX 5050 et 33722 sont des CCSID liés à la page de codes de base 954 sur AIX. Le CCSID signalé par le système d'exploitation est 33722.

3. **Windows** Windows NT utilise la page de codes 932, mais celle-ci est mieux représentée par le CCSID 943. Toutefois, toutes les plateformes de IBM MQ ne prennent pas en charge ce CCSID.

Sous IBM MQ for Windows , le CCSID 932 est utilisé pour représenter la page de codes 932, mais une modification du fichier `./conv/table/ccsid.tbl` peut être effectuée, qui modifie le CCSID utilisé en 943.

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## z/OS

### **z/OS**

Page de codes :

#### **1399**

Ne convertit pas en pages de codes 954, 5035, 5050, 33722

#### **5035**

Ne convertit pas en pages de codes 954, 1399, 5050, 33722

## IBM i

### **IBM i**

Page de codes :

#### **1399**

Ne convertit pas en page de codes 5039

#### **5035**

Ne convertit pas en page de codes 5039

## Japonais Kanji / Katakana Mixte

Détails des CCSID et de la conversion de CCSID pour le japonais Kanji / Katakana Mixed.



Plateforme	CCSID natifs
<b>z/OS</b> z/OS	1390, 5026 (voir la remarque «1», à la page 1008)
<b>IBM i</b> IBM i	5026 (voir la remarque «1», à la page 1008)
<b>AIX</b> AIX	932, 5050, 33722 (voir la remarque «2», à la page 1009)
<b>Windows</b> Windows	932, 943 (voir la remarque «3», à la page 1009)
<b>Linux</b> Linux	943, 5050

### Remarque :

1. **IBM i** **z/OS** Le mode mono-octet des CCSID 1390 et 5026 en EBCDIC contient des caractères minuscules dans des emplacements différents de la présentation standard ou invariante pour le latin de base. Par conséquent, vous devez veiller à ce que les données ne soient pas perdues lorsque les données de message sont converties en autres CCSID. En outre, l'utilisation de ces CCSID comme CCSID par défaut d'un gestionnaire de files d'attente peut entraîner des problèmes lors de la communication avec d'autres gestionnaires de files d'attente. Par exemple, les noms de canal utilisant



des caractères minuscules peuvent ne pas être correctement interprétés sur le système distant. 5026 est un CCSID associé à la page de codes 930. CCSID 5026 est le CCSID signalé sur IBM i lorsque la fonction japonaise Katakana (DBCS) est sélectionnée.

2.  5050 et 33722 sont des CCSID liés à la page de codes de base 954 sur AIX. Le CCSID signalé par le système d'exploitation est 33722.
3.  Windows NT utilise la page de codes 932, mais celle-ci est mieux représentée par le CCSID 943. Toutefois, toutes les plateformes de IBM MQ ne prennent pas en charge ce CCSID.

Sous IBM MQ for Windows, le CCSID 932 est utilisé pour représenter la page de codes 932, mais une modification apportée au fichier `./conv/table/ccsid.tbl` peut modifier le CCSID utilisé en 943.

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## z/OS



Page de codes :

### 1390

Ne convertit pas en pages de codes 954, 5026, 5050, 33722

N'accepte pas les caractères minuscules.

### 5026

Ne convertit pas en pages de codes 954, 1390, 5050, 33722

## IBM i








Page de codes :

### 5026

Ne convertit pas en pages de codes 1390, 5039

## Coréen

Détails des CCSID et de la conversion de CCSID pour le coréen.

<i>Tableau 671. CCSID natifs pour le coréen sur les plateformes prises en charge</i>	
Plateforme	CCSID natifs
 IBM i  z/OS	933, 1364
 AIX  Linux	970
 Windows	949, 1363

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## z/OS



Page de codes :

### 933

Ne convertit pas en page de codes 970






### 1364

Ne convertit pas en page de codes 970


## Chinois simplifié




Détails des CCSID et de la conversion de CCSID pour le chinois simplifié.

Tableau 672. CCSID natifs pour le chinois simplifié sur les plateformes prises en charge


Plateforme	CCSID natifs
 z/OS	935, 1388
 IBM i	935, 1388
 AIX	1383, 1386
 Windows	1381, 1386 (voir note 2)
 Linux	1383

### Remarque :

-  Windows utilise la page de codes 936, mais celle-ci est mieux représentée par le CCSID 1386. Toutefois, toutes les plateformes de IBM MQ ne prennent pas en charge ce CCSID.  
Sous IBM MQ for Windows , le CCSID 1381 est utilisé pour représenter la page de codes 936, mais une modification du fichier `./conv/table/ccsid.tbl` peut être effectuée, qui remplace le CCSID utilisé par 1386.
- IBM MQ prend en charge la norme GB18030 en chinois.

   Sous z/OS, Windows et Linux, la prise en charge de la conversion est assurée entre Unicode (UTF-8 et UTF-16) et CCSID 1388 (EBCDIC avec les extensions GB18030), Unicode (UTF-8 et UTF-16) et CCSID 5488 (GB18030), et entre CCSID 1388 et CCSID 5488.

**Remarque :** Le CCSID doit être défini sur 5488 afin d'utiliser les caractères GB18030 . Toutefois, il n'est pas possible de définir le CCSID sur un gestionnaire de files d'attente créé avec IBM MQ Explorer ou IBM MQ Console. A la place, vous devez créer le gestionnaire de files d'attente à l'aide de l'interface de ligne de commande avec le CCSID 5488 ou utiliser la ligne de commande de l'interface de ligne de commande pour modifier le CCSID après avoir créé le gestionnaire de files d'attente.

 Sous IBM i, le système d'exploitation prend en charge la conversion entre Unicode (UTF-8 et UTF-16) et CCSID 1388 (EBCDIC avec les extensions GB18030).

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## z/OS



Page de codes :

### 935

Ne convertit pas en page de codes 1383






## 1388

Ne convertit pas en page de codes 1383

## chinois traditionnel

Détails des CCSID et de la conversion de CCSID pour le chinois traditionnel.

Tableau 673. CCSID natifs pour le chinois traditionnel sur les plateformes prises en charge

Plateforme	CCSID natifs
 IBM i  z/OS	937
 Windows	950
 AIX  Linux	950, 964

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

## z/OS



Page de codes :

### 937

Ne convertit pas en page de codes 964

### 1388

Ne convertit pas en page de codes 1383

## Linux



Page de codes :

### 964

Ne convertit pas en page de codes 938

## Prise en charge de la conversion z/OS

Liste des conversions CCSID prises en charge.

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS

CCSID	Convertit vers et à partir de CCSIDS
37	256, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 720, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-866, 869-871, 874-875, 880, 897, 903-905, 912, 914-916, 920-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1097, 1100, 1112, 1114-1115, 1122, 1124, 1126, 1130-1132, 1137, 1140-1149, 1200, 1208, 1250-1255, 1257-1258, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5210-5211, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25480, 25617, 25619, 25664, 28709
256	37, 273, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 737, 775, 819, 833, 836, 838, 850, 852, 857, 860-866, 869-871, 875, 880, 905, 1025-1027, 1112, 1122, 1200, 1208, 1251-1252, 1275, 4386, 4929, 4932, 4934, 4946, 4948, 4953, 4960, 4971, 5123, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 13121, 13488, 16804, 17248, 17584, 28709
259	437, 808, 850-852, 855-858, 860-865, 867, 869, 872, 874, 899, 901-902, 915, 1098, 1161-1162, 1200, 1208, 1250-1258, 4946, 4948, 4951-4953, 4960, 4970, 5346, 5348, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584
273	37, 256, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1250, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951-4953, 4960, 4970-4971, 5012, 5123, 5346, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
274	500, 1047
275	37, 437, 500, 819, 850, 1047, 1200, 1208, 1252, 4946, 5348, 8229, 13488, 17584, 28709
277	37, 256, 273, 278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
278	37, 256, 273, 277, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

CCSID	Convertit vers et à partir de CCSIDS
280	37, 256, 273, 277-278, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
281	1047
282	500, 1047, 1200, 1208, 13488, 17584
284	37, 256, 273, 277-278, 280, 285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
285	37, 256, 273, 277-278, 280, 284, 290, 297, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
290	37, 256, 273, 277-278, 280, 284-285, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 895-897, 1009, 1025-1027, 1040-1043, 1047, 1088, 1112, 1122, 1139, 1200, 1208, 1252, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4960, 4992, 5123, 8229, 8482, 9025, 9044, 9049, 9056, 13121, 13488, 17248, 17584, 25473, 25617, 25619, 25664, 28709
293	1200, 1208, 13488, 17584
297	37, 256, 273, 277-278, 280, 284-285, 290, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
300	301, 941, 1200, 1208, 1351, 4396, 8492, 13488, 16684, 17584
301	300, 941, 1200, 1208, 1351, 4396, 8492, 13488, 16684, 17584
367	37, 256, 273, 277-278, 280, 284, 290, 297, 500, 819, 833, 836, 850, 871, 875, 1009, 1026-1027, 1041, 1088, 1115, 1126, 1200, 1208, 4386, 4929, 4932, 4946, 4971, 5123, 5211, 8229, 8482, 9025, 13121, 13488, 17584, 25617, 25664, 28709

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

CCSID	Convertit vers et à partir de CCSIDS
420	37, 256, 424, 437, 500, 720, 737, 775, 819, 850, 852, 857, 860-865, 1008, 1046, 1089, 1098, 1112, 1122, 1127, 1200, 1208, 1252, 1256, 4946, 4948, 4953, 4960, 5104, 5142, 5352, 8229, 8612, 9044, 9049, 9056, 9238, 13488, 16804, 17248, 17584, 28709
423	37, 256, 273, 277-278, 280, 284-285, 297, 437, 500, 737, 775, 813, 819, 838, 850-852, 857, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1009, 1025-1027, 1041-1043, 1112, 1122, 1200, 1208, 1252-1253, 1280, 4909, 4934, 4946, 4948, 4953, 4960, 4970-4971, 5012, 5123, 8229, 9030, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 28709
424	37, 256, 420, 437, 500, 737, 775, 803, 819, 836, 850, 852, 856-857, 860-865, 916, 1112, 1122, 1200, 1208, 1252, 1255, 4932, 4946, 4948, 4952-4953, 4960, 5012, 5351, 8229, 8612, 9044, 9049, 9056, 13488, 16804, 17248, 17584, 28709
437	37, 256, 259, 273, 275, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-863, 865-866, 869-871, 874-875, 880, 897, 903, 905, 912, 914-916, 920-924, 1025-1027, 1040-1043, 1047, 1051, 1097, 1098, 1114-1115, 1126, 1140-1149, 1200, 1208, 1252, 1257, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5210-5211, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 28709
500	37, 256, 273-275, 277-278, 280, 282, 284-285, 290, 297, 367, 420, 423-424, 437, 737, 775, 813, 819, 833, 836, 838, 850-852, 855-858, 860-866, 869-871, 874-875, 880, 891, 895, 897, 903-905, 912, 914-916, 920-924, 1004, 1009-1021, 1023, 1025-1027, 1040-1043, 1046-1047, 1051, 1088-1089, 1097, 1100-1107, 1112, 1114-1115, 1122, 1124-1126, 1129-1133, 1137, 1140-1149, 1200, 1208, 1250-1258, 1275, 1280-1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951-4953, 4960, 4970-4971, 5012, 5123, 5142, 5210-5211, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 9238, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25480, 25617, 25619, 25664, 28709
720	37, 420, 864, 1200, 1208, 1256, 4960, 8229, 8612, 9056, 13488, 16804, 17248, 17584, 28709
737	37, 256, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 833, 836, 838, 850, 869-871, 875, 880, 905, 1025-1027, 1097, 1200, 1208, 1252-1253, 1280, 4386, 4909, 4929, 4932, 4934, 4946, 4971, 5123, 8229, 8482, 8612, 9025, 9030, 9061, 13121, 13488, 16804, 17584, 28709
775	37, 256, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 833, 836, 838, 850, 870-871, 875, 880, 905, 1025-1027, 1097, 1112, 1122, 1200, 1208, 1252, 1257, 4386, 4929, 4932, 4934, 4946, 4971, 5123, 8229, 8482, 8612, 9025, 9030, 13121, 13488, 16804, 17584, 28709
803	424, 819, 850, 856, 862, 916, 1200, 1208, 1252, 1255, 4946, 4952, 5012, 13488, 17584
806	1200, 1208, 13488, 17584
808	259, 858-859, 872, 923-924, 1140, 1148, 1153-1154, 1200, 1208, 5347, 5348, 13488, 17584

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

CCSID	Convertit vers et à partir de CCSIDS
813	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252-1253, 1280, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5349, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
819	37, 256, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 803, 813, 833, 836, 838, 850, 852, 855, 857-858, 860-861, 863-866, 869-871, 874-875, 880, 897, 903, 905, 912, 914-916, 920-924, 1004, 1025-1027, 1041-1043, 1047, 1051, 1088-1089, 1097, 1098, 1112, 1114, 1122-1123, 1126, 1130, 1132, 1137, 1140-1149, 1200, 1208, 1250-1255, 1257-1258, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5210, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
833	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 836, 850, 852, 855, 857, 860-865, 870-871, 891, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1126, 1200, 1208, 1252, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4960, 5123, 8229, 8482, 9025, 9044, 9049, 9056, 13121, 13488, 17248, 17584, 25617, 25619, 25664, 28709
834	926, 951, 1200, 1208, 1362, 4930, 9026, 13488, 17584
835	927, 947, 1200, 1208, 4931, 9027, 13488, 17584, 21427
836	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 424, 437, 500, 737, 775, 819, 833, 850, 852, 855, 857, 870-871, 875, 903, 1009, 1025-1027, 1040-1043, 1088, 1112, 1114-1115, 1122, 1200, 1208, 1252, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4971, 5123, 5210-5211, 8229, 8482, 9025, 9044, 9049, 13121, 13488, 17584, 25479, 25617, 25619, 25664, 28709
837	928, 1200, 1208, 1380, 1385, 4933, 13488, 17584
838	37, 256, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 775, 813, 819, 850, 852, 857, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1112, 1122, 1200, 1208, 1252, 4909, 4934, 4946, 4948, 4953, 4960, 4970-4971, 5012, 5123, 8229, 9030, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 28709
848	924, 1148, 1158, 1200, 1208, 5347, 13488, 17584
849	924, 1148, 1154, 1200, 1208, 5347, 13488, 17584
850	37, 256, 259, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 737, 775, 803, 813, 819, 833, 836, 838, 852, 855-858, 860-866, 869-871, 874-875, 880, 897, 903, 905, 912, 914-916, 920-924, 1004, 1025-1027, 1040-1043, 1047, 1051, 1088-1089, 1097, 1098, 1100, 1112, 1114, 1122, 1126, 1130, 1132, 1140-1149, 1200, 1208, 1250-1257, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951-4953, 4960, 4970-4971, 5012, 5123, 5210, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
851	259, 423, 500, 875, 1200, 1208, 4971, 13488, 17584

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

CCSID	Convertit vers et à partir de CCSIDS
852	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1040-1043, 1047, 1088, 1097, 1200, 1208, 1250, 1252, 1282, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5346, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 25664, 28709
855	37, 259, 273, 277-278, 280, 284-285, 290, 297, 437, 500, 819, 833, 836, 850, 852, 857, 866, 870-871, 878, 880, 912, 915, 1025-1027, 1040-1043, 1088, 1200, 1208, 1250-1252, 1283, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 5123, 5346, 5347, 8229, 8482, 9025, 9044, 9049, 13121, 13488, 17584, 25617, 25619, 25664, 28709
856	259, 273, 424, 500, 803, 850, 862, 916, 1200, 1208, 1255, 4946, 4952, 5012, 5351, 13488, 17584
857	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1040-1043, 1088, 1097, 1200, 1208, 1252, 1254, 1281, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5350, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 25664, 28709
858	37, 259, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 850, 860-861, 865, 871-872, 901-902, 923-924, 1047, 1051, 1140-1149, 1153-1157, 1160-1162, 1164, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
859	808, 872, 901-902, 1153-1157, 1160-1162, 1164, 1200, 1208, 13488, 17584
860	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 838, 850, 852, 857-858, 861, 863, 865, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 923-924, 1025-1027, 1041-1043, 1097, 1140, 1145-1146, 1148, 1200, 1208, 1252, 4386, 4909, 4929, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 28709
861	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 838, 850, 852, 857-858, 860, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 923-924, 1025-1027, 1041-1043, 1097, 1148, 1149, 1200, 1208, 1252, 4386, 4909, 4929, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 28709
862	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 803, 833, 838, 850, 856, 870-871, 875, 880, 905, 916, 1025-1027, 1097, 1200, 1208, 1252, 1255, 4386, 4929, 4934, 4946, 4952, 4971, 5012, 5123, 5351, 8229, 8482, 8612, 9025, 9030, 12712, 13121, 13488, 16804, 17584, 28709



Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

CCSID	Convertit vers et à partir de CCSIDS
863	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 838, 850, 852, 857, 860-861, 865, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1041-1043, 1051, 1097, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 28709
864	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 720, 819, 833, 838, 850, 870-871, 875, 880, 905, 918, 1008, 1025-1027, 1046, 1089, 1097, 1127, 1200, 1208, 1252, 1256, 4386, 4929, 4934, 4946, 4960, 4971, 5104, 5123, 5142, 5352, 8229, 8482, 8612, 9025, 9030, 9056, 9238, 13121, 13488, 16804, 17248, 17584, 28709
865	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 819, 833, 838, 850, 858, 860, 863, 870-871, 875, 880, 905, 923-924, 1025-1027, 1097, 1142-1143, 1148, 1200, 1208, 1252, 4386, 4929, 4934, 4946, 4971, 5123, 5348, 8229, 8482, 8612, 9025, 9030, 13121, 13488, 16804, 17584, 28709
866	37, 256, 437, 500, 819, 850, 855, 870, 878, 880, 915, 1025, 1200, 1208, 1251-1252, 1283, 4946, 4951, 5347, 8229, 13488, 17584, 28709
867	259, 1153-1155, 1160, 1200, 1208, 4899, 5351, 9048, 12712, 13488, 17584
868	918, 1006, 1200, 1208, 13488, 17584
869	37, 256, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 813, 819, 838, 850, 852, 857, 860-861, 863, 870-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252-1254, 1280, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5349, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
870	37, 256, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-866, 869, 871, 874-875, 880, 897, 903, 912, 915-916, 920, 1009, 1025-1027, 1040-1043, 1047, 1088, 1112, 1122, 1200, 1208, 1250, 1252, 1282, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5346, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
871	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869, 870, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
872	259, 808, 858-859, 923-924, 1140-1149, 1153-1155, 1200, 1208, 5347, 5348, 13488, 17584

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

CCSID	Convertit vers et à partir de CCSIDS
874	37, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
875	37, 256, 273, 277-278, 280, 284-285, 297, 367, 423, 437, 500, 737, 775, 813, 819, 836, 838, 850-852, 857, 860-865, 869-871, 874, 880, 897, 903, 912, 916, 920, 1009, 1025-1027, 1041-1043, 1047, 1088, 1112, 1122, 1200, 1208, 1252-1253, 1280, 4909, 4932, 4934, 4946, 4948, 4953, 4960, 4970-4971, 5012, 5123, 5349, 8229, 9030, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
878	855, 866, 880, 915, 1025, 1131, 1200, 1208, 1251, 1283, 4951, 5347, 13488, 17584
880	37, 256, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 775, 813, 819, 838, 850, 852, 855, 857, 860-866, 869-871, 874-875, 878, 897, 903, 912, 915-916, 920, 1009, 1025-1027, 1041-1043, 1112, 1122, 1200, 1208, 1251-1252, 1283, 4909, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5347, 8229, 9030, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 28709
891	500, 833, 1088, 1200, 1208, 4929, 9025, 13121, 13488, 17584, 25664
895	290, 500, 1027, 1041, 1200, 1208, 4386, 5123, 8482, 13488, 17584, 25617
896	290, 1027, 1041, 1200, 1208, 4386, 4992, 5123, 8482, 13488, 17584, 25617
897	37, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 4386, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 8229, 8482, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
899	259
901	259, 858-859, 902, 923-924, 1140, 1148, 1156-1157, 1200, 1208, 5348, 5353, 13488, 17584
902	259, 858-859, 901, 923-924, 1140, 1148, 1156-1157, 1200, 1208, 5348, 5353, 13488, 17584
903	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 836, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 912, 916, 920, 1025-1027, 1041-1043, 1115, 1200, 1208, 1252, 4909, 4932, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5211, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
904	37, 500, 1114, 1200, 1208, 5210, 8229, 13488, 17584, 25480, 28709
905	37, 256, 437, 500, 737, 775, 819, 850, 852, 857, 860-865, 920, 1026, 1112, 1122, 1200, 1208, 1252, 1254, 1281, 4946, 4948, 4953, 4960, 8229, 9044, 9049, 9056, 13488, 17248, 17584, 28709

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

CCSID	Convertit vers et à partir de CCSIDS
912	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 916, 920, 1025-1027, 1041-1043, 1047, 1200, 1208, 1250, 1252, 1282, 4909, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5346, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
914	37, 437, 500, 819, 850, 1200, 1208, 1252, 1257, 4946, 8229, 13488, 17584, 28709
915	37, 259, 437, 500, 819, 850, 855, 866, 870, 878, 880, 1025, 1131, 1200, 1208, 1251-1252, 1283, 4946, 4951, 5347, 8229, 13488, 17584, 28709
916	37, 273, 277-278, 280, 284-285, 297, 423-424, 437, 500, 803, 813, 819, 838, 850, 852, 856-857, 860-863, 869-871, 874-875, 880, 897, 903, 912, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 1255, 4909, 4934, 4946, 4948, 4952-4953, 4970-4971, 5012, 5123, 5351, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
918	864, 868, 1006, 1200, 1208, 4960, 9056, 13488, 17248, 17584
920	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 1025-1026, 1200, 1208, 1252, 1254, 1281, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5350, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 28709
921	37, 437, 500, 819, 850, 922, 1112, 1122, 1200, 1208, 1252, 1257, 4946, 5353, 8229, 13488, 17584, 28709
922	37, 437, 500, 819, 850, 921, 1112, 1122, 1200, 1208, 1252, 1257, 4946, 5353, 8229, 13488, 17584, 28709
923	37, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 850, 858, 860-861, 865, 871-872, 901-902, 924, 1047, 1051, 1140-1149, 1153-1158, 1160-1162, 1164, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
924	37, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 848-850, 858, 860-861, 865, 871-872, 901-902, 923, 1047, 1051, 1140-1149, 1153-1157, 1160-1164, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
926	834, 951, 9026
927	835, 947, 1200, 1208, 4931, 9027, 13488, 17584, 21427
928	837, 1200, 1208, 1380, 13488, 17584
930	931-932, 939, 942-943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
931	930, 932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
932	930-931, 939, 942-943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

<b>CCSID</b>	<b>Convertit vers et à partir de CCSIDS</b>
933	934, 944, 949, 1200, 1208, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13488, 13651, 17317, 17584, 25510, 25520, 25525, 29616, 29621, 33717, 37813
934	933, 949, 5029, 5045, 5460, 9125, 13221, 17317, 25510, 25525, 29621, 33717, 37813
935	936, 946, 1200, 1208, 1381, 1386, 1388, 5031, 5477, 5482, 5484, 9127, 13223, 13488, 17584, 25512
936	935, 946, 1381, 5031, 5477, 5484, 9127, 13223, 25512
937	938, 948, 950, 1200, 1208, 1370, 5033, 5046, 9142, 13488, 17584, 25514, 25524, 29620
938	937, 950, 1370, 5033, 5046, 9142, 25514
939	930-932, 942-943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
941	300-301, 1200, 1208, 1351, 4396, 8492, 13488, 16684, 17584
942	930-932, 939, 943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
943	930-932, 939, 942, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
944	933, 949, 1200, 1208, 5029, 5045, 5460, 9125, 13221, 13488, 17317, 17584, 25520, 25525, 29616, 29621, 33717, 37813
946	935-936, 1200, 1208, 5031, 5484, 9127, 13223, 13488, 17584, 25512
947	835, 927, 1200, 1208, 4931, 9027, 13488, 17584, 21427
948	937, 950, 1200, 1208, 1370, 5033, 5046, 9142, 13488, 17584, 25524, 29620
949	933-934, 944, 1200, 1208, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13488, 13651, 17317, 17584, 25510, 25520, 25525, 29616, 29621, 33717, 37813
950	937-938, 948, 1200, 1208, 1370, 5033, 5046, 9142, 13488, 17584, 25514, 25524, 29620
951	834, 926, 1200, 1208, 1362, 4930, 9026, 13488, 17584
1004	500, 819, 850, 1200, 1208, 4946, 13488, 17584
1006	868, 918, 1200, 1208, 13488, 17584
1008	420, 864, 1200, 1208, 4960, 5104, 8612, 9056, 13488, 16804, 17248, 17584
1009	37, 273, 277-278, 280, 284, 290, 297, 367, 423, 500, 833, 836, 870-871, 875, 880, 1025-1026, 1200, 1208, 4386, 4929, 4932, 4971, 8229, 8482, 9025, 13121, 13488, 17584, 28709
1010	500, 1200, 1208, 13488, 17584

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

CCSID	Convertit vers et à partir de CCSIDS
1011	500, 1200, 1208, 13488, 17584
1 012	500, 1200, 1208, 13488, 17584
1013	500, 1140, 1200, 1208, 13488, 17584
1014	500, 1200, 1208, 13488, 17584
1015	500, 1200, 1208, 13488, 17584
1016	500, 1200, 1208, 13488, 17584
1017	500, 1200, 1208, 13488, 17584
1018	500, 1200, 1208, 13488, 17584
1019	500, 1200, 1208, 13488, 17584
1020	500
1021	500
1023	500
1025	37, 256, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-866, 869-871, 874-875, 878, 880, 897, 903, 912, 915-916, 920, 1009, 1026-1027, 1040-1043, 1051, 1088, 1112, 1122, 1131, 1200, 1208, 1251-1252, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5347, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
1026	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-865, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1009, 1025, 1027, 1040-1043, 1047, 1088, 1112, 1122, 1200, 1208, 1252, 1254, 1281, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5350, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
1027	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-865, 869-871, 874-875, 880, 895-897, 903, 912, 916, 1025-1026, 1040-1043, 1047, 1088, 1112, 1122, 1139, 1200, 1208, 1252, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 4992, 5012, 5123, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
1040	37, 273, 277-278, 280, 284-285, 290, 297, 437, 500, 833, 836, 850, 852, 855, 857, 870-871, 1025-1027, 1041-1043, 1088, 1200, 1208, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 5123, 8229, 8482, 9025, 9044, 9049, 13121, 13488, 17584, 25617, 25619, 25664, 28709
1041	37, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 895-897, 903, 912, 916, 1025-1027, 1040, 1042-1043, 1088, 1200, 1208, 1252, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 4992, 5012, 5123, 8229, 8482, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 17584, 25473, 25479, 25617, 25619, 25664, 28709

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

CCSID	Convertit vers et à partir de CCSIDS
1042	37, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 1025-1027, 1040, 1041, 1043, 1088, 1200, 1208, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 8229, 8482, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 17584, 25473, 25479, 25617, 25619, 25664, 28709
1043	37, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 1025-1027, 1040, 1041, 1042, 1088, 1114, 1200, 1208, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5210, 8229, 8482, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 17584, 25473, 25479, 25617, 25619, 25664, 28709
1046	420, 500, 864, 1089, 1127, 1200, 1208, 1256, 4960, 5142, 5352, 8612, 9056, 9238, 13488, 16804, 17248, 17584
1047	37, 273-275, 277-278, 280, 281, 282, 284-285, 290, 297, 437, 500, 819, 850, 852, 858, 870-871, 875, 912, 923-924, 1026-1027, 1140-1149, 1200, 1208, 1252, 1254, 4946, 4948, 5123, 8229, 8482, 13488, 17584, 28709
1051	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 871, 923-924, 1025, 1097, 1140-1149, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1088	37, 273, 277-278, 280, 284-285, 290, 297, 367, 500, 819, 833, 836, 850, 852, 855, 857, 870-871, 875, 891, 1025-1027, 1040-1043, 1126, 1200, 1208, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4971, 5123, 8229, 8482, 9025, 9044, 9049, 13121, 13488, 17584, 25617, 25619, 25664, 28709
1089	420, 500, 819, 850, 864, 1046, 1127, 1200, 1208, 1256, 4946, 4960, 5142, 5352, 8612, 9056, 9238, 13488, 16804, 17248, 17584
1097	37, 437, 500, 737, 775, 819, 850, 852, 857, 860-865, 1051, 1098, 1112, 1122, 1200, 1208, 1252, 4946, 4948, 4953, 4960, 8229, 9044, 9049, 9056, 13488, 17248, 17584, 28709
1098	259, 420, 437, 819, 850, 1097, 1200, 1208, 1252, 4946, 8612, 13488, 16804, 17584
1100	37, 273, 277-278, 280, 284-285, 297, 500, 850, 4946, 8229, 28709
1101	500
1102	500
1103	500
1104	500
1105	500
1106	500
1107	500

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

<b>CCSID</b>	<b>Convertit vers et à partir de CCSIDS</b>
1112	37, 256, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 775, 819, 833, 836, 838, 850, 870-871, 875, 880, 905, 921-922, 1025-1027, 1097, 1122, 1200, 1208, 1252, 1257, 4386, 4929, 4932, 4934, 4946, 4971, 5123, 5353, 8229, 8482, 8612, 9025, 9030, 13121, 13488, 16804, 17584, 28709
1114	37, 437, 500, 819, 836, 850, 904, 1043, 1115, 1200, 1208, 4932, 4946, 5210-5211, 8229, 13488, 17584, 25480, 25619, 28709
1115	37, 367, 437, 500, 836, 903, 1114, 1200, 1208, 4932, 5210-5211, 8229, 13488, 17584, 25479, 28709
1122	37, 256, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 775, 819, 833, 836, 838, 850, 870-871, 875, 880, 905, 921-922, 1025-1027, 1097, 1112, 1200, 1208, 1252, 1257, 4386, 4929, 4932, 4934, 4946, 4971, 5123, 5353, 8229, 8482, 8612, 9025, 9030, 13121, 13488, 16804, 17584, 28709
1123	819, 1124-1125, 1148, 1200, 1208, 1251-1252, 1283, 5347, 13488, 17584
1124	37, 500, 1123, 1125, 1200, 1208, 1251, 1283, 5347, 8229, 13488, 17584, 28709
1125	500, 1123, 1124, 1200, 1208, 1251, 1283, 5347, 13488, 17584
1126	37, 367, 437, 500, 819, 833, 850, 1088, 1200, 1208, 1252, 4929, 4946, 8229, 9025, 13121, 13488, 17584, 25664, 28709
1127	420, 864, 1046, 1089, 1256, 4960, 5142, 8612, 9056, 9238, 16804, 17248
1129	500, 1130, 1200, 1208, 1258, 5354, 13488, 17584
1130	37, 500, 819, 850, 1129, 1200, 1208, 1252, 1258, 4946, 5354, 8229, 13488, 17584, 28709
1131	37, 500, 878, 915, 1025, 1200, 1208, 1251, 1283, 5347, 8229, 13488, 17584, 28709
1132	37, 500, 819, 850, 1133, 1200, 1208, 1252, 4946, 8229, 13488, 17584, 28709
1133	500, 1132, 1200, 1208, 13488, 17584
1137	37, 500, 819, 1200, 1208, 8229, 13488, 17584, 28709
1139	290, 1027, 4386, 5123, 8482
1140	37, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 850, 858, 860, 863, 871-872, 901-902, 923-924, 1013, 1047, 1051, 1141-1149, 1153-1157, 1160-1162, 1164, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1141	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 871-872, 923-924, 1047, 1051, 1140, 1142-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

CCSID	Convertit vers et à partir de CCSIDS
1142	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 865, 871-872, 923-924, 1047, 1051, 1140-1141, 1143-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1143	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 865, 871-872, 923-924, 1047, 1051, 1140-1142, 1144-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1144	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 871-872, 923-924, 1047, 1051, 1140-1143, 1145-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1145	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 860, 863, 871-872, 923-924, 1047, 1051, 1140-1144, 1146-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1146	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 860, 863, 871-872, 923-924, 1047, 1051, 1140-1145, 1147-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1147	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 871-872, 923-924, 1047, 1051, 1140-1146, 1148-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1148	37, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 848-850, 858, 860-861, 863, 865, 871-872, 901-902, 923-924, 1047, 1051, 1123, 1140-1147, 1149, 1153-1164, 1200, 1208, 1252, 1275, 4899, 4946, 5348, 5349, 8229, 12712, 13488, 17584, 28709
1149	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 861, 863, 871-872, 923-924, 1047, 1051, 1140-1148, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1153	808, 858-859, 867, 872, 923-924, 1140-1149, 1154-1157, 1160-1162, 1200, 1208, 5348, 9044, 13488, 17584
1154	808, 849, 858-859, 867, 872, 923-924, 1140-1149, 1153, 1155-1157, 1160-1162, 1200, 1208, 5347, 5348, 13488, 17584
1155	858-859, 867, 872, 923-924, 1140-1149, 1153-1154, 1156-1157, 1160-1162, 1200, 1208, 5348, 5350, 9049, 13488, 17584
1156	858-859, 901-902, 923-924, 1140-1149, 1153-1155, 1157, 1160, 1200, 1208, 5348, 5353, 12712, 13488, 17584
1157	858-859, 901-902, 923-924, 1140-1149, 1153-1156, 1160, 1200, 1208, 5348, 5353, 12712, 13488, 17584
1158	848, 923, 1148, 1200, 1208, 5347, 5348, 13488, 17584
1159	1148, 1200, 1208, 13488, 17584
1160	858-859, 867, 923-924, 1140-1149, 1153-1157, 1161-1162, 1200, 1208, 5348, 13488, 17584



Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

CCSID	Convertit vers et à partir de CCSIDS
1161	259, 858-859, 923-924, 1140-1149, 1153-1155, 1160, 5348, 17584
1162	259, 858-859, 923-924, 1140-1149, 1153-1155, 1160, 5348, 17584
1163	924, 1148, 1164, 5354, 17584
1164	858-859, 923-924, 1140, 1148, 1163, 1200, 1208, 5348, 5354, 13488, 17584
1166	1200,1208,13488,17584
1 200	37, 256, 259, 273, 275, 277-278, 280, 282, 284-285, 290, 293, 297, 300-301, 367, 420, 423-424, 437, 500, 720, 737, 775, 803, 806, 808, 813, 819, 833-838, 848-852, 855-872, 874-875, 878, 880, 891, 895-897, 901-905, 912, 914-916, 918, 920-924, 927-928, 930, 932-933, 935, 937, 939, 941-944, 946-951, 1004, 1006, 1008-1019, 1025-1027, 1040-1043, 1046-1047, 1051, 1088-1089, 1097-1098, 1112, 1114-1115, 1122-1126, 1129-1133, 1137, 1140-1149, 1153-1160, 1164, 1166, 1208, 1250-1258, 1275-1277, 1280-1285, 1351, 1362-1364, 1370-1371, 1374-1379, 1380-1381, 1385-1386, 1388, 1390, 1399, 4899, 4909, 4930, 4933, 4948, 4951-4952, 4960, 4971, 5012, 5039, 5104, 5123, 5142, 5210, 5346-5354, 8482, 8612, 9027, 9030, 9044, 9048-9049, 9056, 9061, 9066, 9238, 12712, 13121, 13218, 13488, 16684, 16804, 17248, 17584, 21427, 28709
1208	37, 256, 259, 273, 275, 277-278, 280, 282, 284-285, 290, 293, 297, 300-301, 367, 420, 423-424, 437, 500, 720, 737, 775, 803, 806, 808, 813, 819, 833-838, 848-852, 855-872, 874-875, 878, 880, 891, 895-897, 901-905, 912, 914-916, 918, 920-924, 927-928, 930, 932-933, 935, 937, 939, 941-944, 946-951, 1004, 1006, 1008-1019, 1025-1027, 1040-1043, 1046-1047, 1051, 1088-1089, 1097-1098, 1112, 1114-1115, 1122-1126, 1129-1133, 1137, 1140-1149, 1153-1160, 1164, 1166, 1200, 1250-1258, 1275-1277, 1280-1285, 1351, 1362-1364, 1370-1371, 1374-1379, 1380-1381, 1385-1386, 1388, 1390, 1399, 4899, 4909, 4930, 4933, 4948, 4951-4952, 4960, 4971, 5012, 5026, 5035, 5039, 5104, 5123, 5142, 5210, 5346-5354, 8482, 8612, 9027, 9030, 9044, 9048-9049, 9056, 9061, 9066, 9238, 12712, 13121, 13218, 13488, 16684, 16804, 17248, 17584, 21427, 28709
1250	37, 259, 273, 500, 819, 850, 852, 855, 870, 912, 1200, 1208, 1252, 1282, 4946, 4948, 4951, 5346, 8229, 9044, 13488, 17584, 28709
1251	37, 256, 259, 500, 819, 850, 855, 866, 878, 880, 915, 1025, 1123-1125, 1131, 1200, 1208, 1252, 1283, 4946, 4951, 5347, 8229, 13488, 17584, 28709
1252	37, 256, 259, 273, 275, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 737, 775, 803, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-866, 869-871, 874-875, 880, 897, 903, 905, 912, 914-916, 920-924, 1025-1027, 1041, 1047, 1051, 1097-1098, 1112, 1122-1123, 1126, 1130, 1132, 1140-1149, 1200, 1208, 1250-1251, 1254-1255, 1257, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25617, 28709
1253	37, 259, 423, 500, 737, 813, 819, 850, 869, 875, 1200, 1208, 1280, 4909, 4946, 4971, 5349, 8229, 9061, 13488, 17584, 28709

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

<b>CCSID</b>	<b>Convertit vers et à partir de CCSIDS</b>
1254	37, 259, 500, 819, 850, 857, 869, 905, 920, 1026, 1047, 1200, 1208, 1252, 1281, 4946, 4953, 5350, 8229, 9049, 9061, 13488, 17584, 28709
1255	37, 259, 424, 500, 803, 819, 850, 856, 862, 916, 1200, 1208, 1252, 1281, 4946, 4952, 5012, 5351, 8229, 13488, 17584, 28709
1256	259, 420, 500, 720, 850, 864, 1046, 1089, 1127, 1200, 1208, 4946, 4960, 5142, 5352, 8612, 9056, 9238, 13488, 16804, 17248, 17584
1257	37, 259, 437, 500, 775, 819, 850, 914, 921-922, 1112, 1122, 1200, 1208, 1252, 4946, 5353, 8229, 13488, 17584, 28709
1258	37, 259, 500, 819, 1129-1130, 1200, 1208, 5354, 8229, 13488, 17584, 28709
1275	37, 256, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 871, 923-924, 1051, 1140-1149, 1200, 1208, 1252, 4946, 5348, 8229, 13488, 17584, 28709
1276	1200, 1208, 13488, 17584
1277	1200, 1208, 13488, 17584
1280	37, 423, 437, 500, 737, 813, 819, 850, 869, 875, 1200, 1208, 1252-1253, 4909, 4946, 4971, 5349, 8229, 9061, 13488, 17584, 28709
1281	37, 437, 500, 819, 850, 857, 905, 920, 1026, 1200, 1208, 1252, 1254-1255, 4946, 4953, 5350, 8229, 9049, 13488, 17584, 28709
1282	500, 852, 870, 912, 1200, 1208, 1250, 4948, 5346, 9044, 13488, 17584
1283	37, 437, 500, 819, 850, 855, 866, 878, 880, 915, 1025, 1123-1125, 1131, 1200, 1208, 1251-1252, 4946, 4951, 5347, 8229, 13488, 17584, 28709
1284	1200, 1208, 13488, 17584
1285	1200, 1208, 13488, 17584
1351	300-301, 941, 1200, 1208, 4396, 8492, 13488, 16684, 17584
1362	834, 951, 1200, 1208, 4930, 9026, 13488, 17584
1363	933, 949, 1200, 1208, 1364, 5029, 5045, 5460, 9125, 9555, 13221, 13488, 13651, 17317, 17584, 25525, 29621, 33717, 37813
1364	933, 949, 1200, 1208, 1363, 5029, 5045, 5460, 9125, 9555, 13221, 13488, 13651, 17317, 17584, 25525, 29621, 33717, 37813
1370	937-938, 948, 950, 1200, 1208, 1371, 5033, 5046, 9142, 13488, 17584, 25514, 25524, 29620
1371	1200, 1208, 1370, 13488, 17584
1374	1200, 1208
1375	1200, 1208
1376	1200, 1208
1377	1200, 1208
1378	1200, 1208
1379	1200, 1208

<i>Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)</i>	
<b>CCSID</b>	<b>Convertit vers et à partir de CCSIDS</b>
1380	837, 928, 1200, 1208, 1385, 4933, 13488, 17584
1381	935-936, 1200, 1208, 1386, 1388, 5031, 5477, 5482, 5484, 9127, 13223, 13488, 17584, 25512
1385	837, 1200, 1208, 1380, 4933, 13488, 17584
1386	935, 1200, 1208, 1381, 1388, 5031, 5477, 5482, 5484, 9127, 13223, 13488, 17584
1388	935, 1200, 1208, 1381, 1386, 5031, 5477, 5482, 5484, 5488, 9127, 13223, 13488, 17584
1390	930-932, 939, 942-943, 1200, 1208, 1399, 5026, 5028, 5035, 5038-5039, 5055, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
1399	930-932, 939, 942-943, 1200, 1208, 1390, 5026, 5028, 5035, 5038-5039, 5050, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
4386	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 895-897, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1139, 1252, 4929, 4932, 4946, 4948, 4951, 4953, 4960, 4992, 5123, 8229, 8482, 9025, 9044, 9049, 9056, 13121, 17248, 25473, 25617, 25619, 25664, 28709
4396	300-301, 941, 1351, 8492, 16684
4899	867, 1148, 1200, 1208, 5351, 9048, 12712, 13488, 17584
4909	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252-1253, 1280, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5349, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
4929	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 891, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1126, 1252, 4386, 4932, 4946, 4948, 4951, 4953, 4960, 5123, 8229, 8482, 9025, 9044, 9049, 9056, 13121, 17248, 25617, 25619, 25664, 28709
4930	834, 951, 1200, 1208, 1362, 9026, 13488, 17584
4931	835, 927, 947, 9027, 21427
4932	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 424, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 870-871, 875, 903, 1009, 1025-1027, 1040-1043, 1088, 1112, 1114-1115, 1122, 1252, 4386, 4929, 4946, 4948, 4951, 4953, 4971, 5123, 5210-5211, 8229, 8482, 9025, 9044, 9049, 13121, 25479, 25617, 25619, 25664, 28709
4933	837, 1200, 1208, 1380, 1385, 13488, 17584
4934	37, 256, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 775, 813, 819, 838, 850, 852, 857, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1112, 1122, 1252, 4909, 4946, 4948, 4953, 4960, 4970-4971, 5012, 5123, 8229, 9030, 9044, 9049, 9056, 9061, 9066, 17248, 25473, 25479, 25617, 25619, 28709

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

CCSID	Convertit vers et à partir de CCSIDS
4946	37, 256, 259, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 737, 775, 803, 813, 819, 833, 836, 838, 850, 852, 855-858, 860-866, 869-871, 874-875, 880, 897, 903, 905, 912, 914-916, 920-924, 1004, 1025-1027, 1040-1043, 1047, 1051, 1088-1089, 1097-1098, 1100, 1112, 1114, 1122, 1126, 1130, 1132, 1140-1149, 1250-1257, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4948, 4951-4953, 4960, 4970-4971, 5012, 5123, 5210, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 16804, 17248, 25473, 25479, 25617, 25619, 25664, 28709
4948	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1040-1043, 1047, 1088, 1097, 1200, 1208, 1250, 1252, 1282, 4386, 4909, 4929, 4932, 4934, 4946, 4951, 4953, 4970-4971, 5012, 5123, 5346, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 25664, 28709
4951	37, 259, 273, 277-278, 280, 284-285, 290, 297, 437, 500, 819, 833, 836, 850, 852, 855, 857, 866, 870-871, 878, 880, 912, 915, 1025-1027, 1040-1043, 1088, 1200, 1208, 1250-1252, 1283, 4386, 4929, 4932, 4946, 4948, 4953, 5123, 5346, 5347, 8229, 8482, 9025, 9044, 9049, 13121, 13488, 17584, 25617, 25619, 25664, 28709
4952	259, 273, 424, 500, 803, 850, 856, 862, 916, 1200, 1208, 1255, 4946, 5012, 5351, 13488, 17584
4953	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1040-1043, 1088, 1097, 1252, 1254, 1281, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4970-4971, 5012, 5123, 5350, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 16804, 25473, 25479, 25617, 25619, 25664, 28709
4960	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 720, 819, 833, 838, 850, 864, 870-871, 875, 880, 905, 918, 1008, 1025-1027, 1046, 1089, 1097, 1127, 1200, 1208, 1252, 1256, 4386, 4929, 4934, 4946, 4971, 5104, 5123, 5142, 5352, 8229, 8482, 8612, 9025, 9030, 9056, 9238, 13121, 13488, 16804, 17248, 17584, 28709
4970	37, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1252, 4909, 4934, 4946, 4948, 4953, 4971, 5012, 5123, 8229, 9030, 9044, 9049, 9061, 9066, 25473, 25479, 25617, 25619, 28709
4971	37, 256, 273, 277-278, 280, 284-285, 297, 367, 423, 437, 500, 737, 775, 813, 819, 836, 838, 850-852, 857, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1009, 1025-1027, 1041-1043, 1047, 1088, 1112, 1122, 1200, 1208, 1252-1253, 1280, 4909, 4932, 4934, 4946, 4948, 4953, 4960, 4970, 5012, 5123, 5349, 8229, 9030, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
4992	290, 896, 1027, 1041, 4386, 5123, 8482, 25617

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

CCSID	Convertit vers et à partir de CCSIDS
5012	37, 273, 277-278, 280, 284-285, 297, 423-424, 437, 500, 803, 813, 819, 838, 850, 852, 856-857, 860-863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 1255, 4909, 4934, 4946, 4948, 4952-4953, 4970-4971, 5123, 5351, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
5026	930-932, 939, 942-943, 1390, 1399, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 1208, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
5028	930-932, 939, 942-943, 1390, 1399, 5026, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
5029	933-934, 944, 949, 1363-1364, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813
5031	935-936, 946, 1381, 1386, 1388, 5477, 5482, 5484, 9127, 13223, 25512
5033	937-938, 948, 950, 1370, 5046, 9142, 25514, 25524, 29620
5035	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5038-5039, 9122, 9124, 9131, 9135, 1208, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
5038	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
5039	930-932, 939, 942-943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
5045	933-934, 944, 949, 1363-1364, 5029, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813
5046	937-938, 948, 950, 1370, 5033, 9142, 25514, 25524, 29620
5104	420, 864, 1008, 1200, 1208, 4960, 8612, 9056, 13488, 16804, 17248, 17584
5123	290, 367, 423, 437, 819, 1027, 1041, 1047, 1140-1149, 1156, 1157, 1160, 1200, 1208, 1252, 4948, 5348, 8482, 13488
5142	420, 500, 864, 1046, 1089, 1127, 1200, 1208, 1256, 4960, 5352, 8612, 9056, 9238, 13488, 16804, 17248, 17584
5210	37, 437, 500, 819, 836, 850, 904, 1043, 1114-1115, 1200, 1208, 4932, 4946, 5211, 8229, 13488, 17584, 25480, 25619, 28709
5211	37, 367, 437, 500, 836, 903, 1114-1115, 4932, 5210, 8229, 25479, 28709
5346	37, 259, 273, 500, 819, 850, 852, 855, 870, 912, 1200, 1208, 1250, 1252, 1282, 4946, 4948, 4951, 8229, 9044, 13488, 17584, 28709
5347	808, 848-849, 855, 866, 872, 878, 880, 915, 1025, 1123-1125, 1131, 1154, 1158, 1200, 1208, 1251, 1283, 4951, 13488, 17584

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

CCSID	Convertit vers et à partir de CCSIDS
5348	37, 259, 273, 275, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 850, 858, 860-861, 863, 865, 871-872, 901-902, 923-924, 1051, 1140-1149, 1153-1158, 1160-1162, 1164, 1200, 1208, 1252, 1275, 4946, 8229, 13488, 17584, 28709
5349	813, 869, 875, 1148, 1200, 1208, 1253, 1280, 4909, 4971, 9061, 13488, 17584
5350	857, 920, 1026, 1155, 1200, 1208, 1254, 1281, 4953, 9049, 13488, 17584
5351	424, 856, 862, 867, 916, 1200, 1208, 1255, 4899, 4952, 5012, 9048, 12712, 13488, 17584
5352	420, 864, 1046, 1089, 1200, 1208, 1256, 4960, 5142, 8612, 9056, 9238, 13488, 16804, 17248, 17584
5353	901-902, 921-922, 1112, 1122, 1156-1157, 1200, 1208, 1257, 13488, 17584
5354	1129-1130, 1163, 1164, 1200, 1208, 1258, 13488, 17584
5460	933-934, 944, 949, 1363-1364, 5029, 5045, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813
5477	935-936, 1381, 1386, 1388, 5031, 5482, 5484, 9127, 13223, 25512
5482	935, 1381, 1386, 1388, 5031, 5477, 5484, 9127, 13223
5484	935-936, 946, 1381, 1386, 1388, 5031, 5477, 5482, 9127, 13223, 25512
5488	1388
8229	37, 256, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 720, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-866, 869-871, 874-875, 880, 897, 903-905, 912, 914-916, 920-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1097, 1100, 1112, 1114-1115, 1122, 1124, 1126, 1130-1132, 1137, 1140-1149, 1250-1255, 1257-1258, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5210-5211, 5346, 5348, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 16804, 17248, 25473, 25479, 25480, 25617, 25619, 25664, 28709
8482	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 895-897, 1009, 1025-1027, 1040-1043, 1047, 1088, 1112, 1122, 1139, 1200, 1208, 1252, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4960, 4992, 5123, 8229, 9025, 9044, 9049, 9056, 13121, 13488, 17248, 17584, 25473, 25617, 25619, 25664, 28709
8492	300-301, 941, 1351, 4396, 16684
8612	37, 256, 420, 424, 437, 500, 720, 737, 775, 819, 850, 852, 857, 860-865, 1008, 1046, 1089, 1098, 1112, 1122, 1127, 1200, 1208, 1252, 1256, 4946, 4948, 4953, 4960, 5104, 5142, 5352, 8229, 9044, 9049, 9056, 9238, 13488, 16804, 17248, 17584, 28709

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

CCSID	Convertit vers et à partir de CCSIDS
9025	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 891, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1126, 1252, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4960, 5123, 8229, 8482, 9044, 9049, 9056, 13121, 17248, 25617, 25619, 25664, 28709
9026	834, 926, 951, 1362, 4930
9027	835, 927, 947, 1200, 1208, 4931, 13488, 17584, 21427
9030	37, 256, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 775, 813, 819, 838, 850, 852, 857, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1112, 1122, 1200, 1208, 1252, 4909, 4934, 4946, 4948, 4953, 4960, 4970-4971, 5012, 5123, 8229, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 28709
9044	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1040-1043, 1047, 1088, 1097, 1153, 1200, 1208, 1250, 1252, 1282, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5346, 8229, 8482, 8612, 9025, 9030, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 25664, 28709
9048	867, 1200, 1208, 4899, 5351, 12712, 13488, 17584
9049	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1040-1043, 1088, 1097, 1155, 1200, 1208, 1252, 1254, 1281, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5350, 8229, 8482, 8612, 9025, 9030, 9044, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 25664, 28709
9056	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 720, 819, 833, 838, 850, 864, 870-871, 875, 880, 905, 918, 1008, 1025-1027, 1046, 1089, 1097, 1127, 1200, 1208, 1252, 1256, 4386, 4929, 4934, 4946, 4960, 4971, 5104, 5123, 5142, 5352, 8229, 8482, 8612, 9025, 9030, 9238, 13121, 13488, 16804, 17248, 17584, 28709
9061	37, 256, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252-1254, 1280, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5349, 8229, 9030, 9044, 9049, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
9066	37, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 8229, 9030, 9044, 9049, 9061, 13488, 17584, 25473, 25479, 25617, 25619, 28709
9122	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

<b>CCSID</b>	<b>Convertit vers et à partir de CCSIDS</b>
9124	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
9125	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813
9127	935-936, 946, 1381, 1386, 1388, 5031, 5477, 5482, 5484, 13223, 25512
9131	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
9135	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
9142	937-938, 948, 950, 1370, 5033, 5046, 25514, 25524, 29620
9238	420, 500, 864, 1046, 1089, 1127, 1200, 1208, 1256, 4960, 5142, 5352, 8612, 9056, 13488, 16804, 17248, 17584
9555	933, 949, 1363-1364, 5029, 5045, 5460, 9125, 13221, 13651, 17317, 25525, 29621, 33717, 37813
12712	862, 867, 1148, 1156-1157, 1200, 1208, 4899, 5351, 9048, 13488, 17584
13121	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 891, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1126, 1200, 1208, 1252, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4960, 5123, 8229, 8482, 9025, 9044, 9049, 9056, 13488, 17248, 17584, 25617, 25619, 25664, 28709
13218	930-932, 939, 942-943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
13219	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
13221	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813
13223	935-936, 946, 1381, 1386, 1388, 5031, 5477, 5482, 5484, 9127, 25512
13231	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 17314, 25508, 25518, 29614, 33698-33700, 37796



Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

CCSID	Convertit vers et à partir de CCSIDS
13488	37, 256, 259, 273, 275, 277-278, 280, 282, 284-285, 290, 293, 297, 300-301, 367, 420, 423-424, 437, 500, 720, 737, 775, 803, 806, 808, 813, 819, 833-838, 848-852, 855-872, 874-875, 878, 880, 891, 895-897, 901-905, 912, 914-916, 918, 920-924, 927-928, 930, 932-933, 935, 937, 939, 941-944, 946-951, 1004, 1006, 1008-1019, 1025-1027, 1040-1043, 1046-1047, 1051, 1088-1089, 1097-1098, 1112, 1114-1115, 1122-1126, 1129-1133, 1137, 1140-1149, 1153-1160, 1164, 1166, 1200, 1208, 1250-1258, 1275-1277, 1280-1285, 1351, 1362-1364, 1370-1371, 1380-1381, 1385-1386, 1388, 1390, 1399, 4899, 4909, 4930, 4933, 4948, 4951-4952, 4960, 4971, 5012, 5039, 5104, 5123, 5142, 5210, 5346-5354, 8482, 8612, 9027, 9030, 9044, 9048-9049, 9056, 9061, 9066, 9238, 12712, 13121, 13218, 16684, 16804, 17248, 17584, 21427, 28709
13651	933, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 17317, 25525, 29621, 33717, 37813
16684	300-301, 941, 1200, 1208, 1351, 4396, 8492, 13488, 17584
16804	37, 256, 420, 424, 437, 500, 720, 737, 775, 819, 850, 852, 857, 860-865, 1008, 1046, 1089, 1098, 1112, 1122, 1127, 1200, 1208, 1252, 1256, 4946, 4948, 4953, 4960, 5104, 5142, 5352, 8229, 8612, 9044, 9049, 9056, 9238, 13488, 17248, 17584, 28709
17248	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 720, 819, 833, 838, 850, 864, 870-871, 875, 880, 905, 918, 1008, 1025-1027, 1046, 1089, 1097, 1127, 1200, 1208, 1252, 1256, 4386, 4929, 4934, 4946, 4960, 4971, 5104, 5123, 5142, 5352, 8229, 8482, 8612, 9025, 9030, 9056, 9238, 13121, 13488, 16804, 17584, 28709
17314	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 25508, 25518, 29614, 33698-33700, 37796
17317	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 25510, 25520, 25525, 29616, 29621, 33717, 37813
17584	37, 256, 259, 273, 275, 277-278, 280, 282, 284-285, 290, 293, 297, 300-301, 367, 420, 423-424, 437, 500, 720, 737, 775, 803, 806, 808, 813, 819, 833-838, 848-852, 855-872, 874-875, 878, 880, 891, 895-897, 901-905, 912, 914-916, 918, 920-924, 927-928, 930, 932-933, 935, 937, 939, 941-944, 946-951, 1004, 1006, 1008-1019, 1025-1027, 1040-1043, 1046-1047, 1051, 1088-1089, 1097-1098, 1112, 1114-1115, 1122-1126, 1129-1133, 1137, 1140-1149, 1153-1160, 1164, 1166, 1200, 1208, 1250-1258, 1275-1277, 1280-1285, 1351, 1362-1364, 1370-1371, 1380-1381, 1385-1386, 1388, 1390, 1399, 4899, 4909, 4930, 4933, 4948, 4951-4952, 4960, 4971, 5012, 5039, 5104, 5123, 5142, 5210, 5346-5354, 8482, 8612, 9027, 9030, 9044, 9048-9049, 9056, 9061, 9066, 9238, 12712, 13121, 13218, 13488, 16684, 16804, 17248, 21427, 28709
21427	835, 927, 947, 1200, 1208, 4931, 9027, 13488, 17584
25473	37, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1252, 4386, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 8229, 8482, 9030, 9044, 9049, 9061, 9066, 25479, 25617, 25619, 28709

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

CCSID	Convertit vers et à partir de CCSIDS
25479	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 836, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1115, 1252, 4909, 4932, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5211, 8229, 9030, 9044, 9049, 9061, 9066, 25473, 25617, 25619, 28709
25480	37, 500, 904, 1114, 5210, 8229, 28709
25508	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25518, 29614, 33698-33700, 37796
25510	933-934, 949, 5029, 5045, 5460, 9125, 13221, 17317, 25525, 29621, 33717, 37813
25512	935-936, 946, 1381, 5031, 5477, 5484, 9127, 13223
25514	937-938, 950, 1370, 5033, 5046, 9142
25518	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 29614, 33698-33700, 37796
25520	933, 944, 949, 5029, 5045, 5460, 9125, 13221, 17317, 25525, 29616, 29621, 33717, 37813
25524	937, 948, 950, 1370, 5033, 5046, 9142, 29620
25525	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 29616, 29621, 33717, 37813
25617	37, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 895-897, 903, 912, 916, 1025-1027, 1040-1043, 1088, 1252, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 4992, 5012, 5123, 8229, 8482, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 25473, 25479, 25619, 25664, 28709
25619	37, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 1025-1027, 1040-1043, 1088, 1114, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5210, 8229, 8482, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 25473, 25479, 25617, 25664, 28709
25664	37, 273, 277-278, 280, 284-285, 290, 297, 367, 500, 819, 833, 836, 850, 852, 855, 857, 870-871, 875, 891, 1025-1027, 1040-1043, 1088, 1126, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4971, 5123, 8229, 8482, 9025, 9044, 9049, 13121, 25617, 25619, 28709

Tableau 674. Prise en charge de la conversion CCSID IBM MQ for z/OS (suite)

CCSID	Convertit vers et à partir de CCSIDS
28709	37, 256, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 720, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-866, 869-871, 874-875, 880, 897, 903-905, 912, 914-916, 920-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1097, 1100, 1112, 1114-1115, 1122, 1124, 1126, 1130-1132, 1137, 1140-1149, 1200, 1208, 1250-1255, 1257-1258, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5210-5211, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25480, 25617, 25619, 25664
29614	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 33698-33700, 37796
29616	933, 944, 949, 5029, 5045, 5460, 9125, 13221, 17317, 25520, 25525, 29621, 33717, 37813
29620	937, 948, 950, 1370, 5033, 5046, 9142, 25524
29621	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 33717, 37813
33698	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33699-33700, 37796
33699	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698, 33700, 37796
33700	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33699, 37796
33717	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 37813
37796	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700
37813	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717

## **Prise en charge de la conversion IBM i**

La liste complète des CCSID et des conversions prises en charge par IBM i est disponible dans la publication IBM i appropriée.

Les pages de codes prises en charge sont répertoriées dans [Mappages de CCSID pris en charge](#).

## **Prise en charge de la conversion Unicode**

Certaines plateformes prennent en charge la conversion des données utilisateur vers ou depuis le codage Unicode. Les deux formes de codage Unicode prises en charge sont UTF-16 (CCSID 1200, 13488 et 17584) et UTF-8 (CCSID 1208). Vous devez utiliser les CCSID 1200 ou 1208, car ils représentent la version Unicode la plus récente prise en charge.

Les paires de substitution UTF-16 (une paire de caractères UTF-16 de 2 octets compris entre X'D800'et X'DFFF'représentant un point de code Unicode au-dessus de U + FFFF) sont prises en charge. Si un CCSID cible ne contient pas de mappage pour un point de code représenté par une paire de substitution UTF-16 , la paire de caractères est convertie en un seul caractère de substitution.

La combinaison des séquences de caractères est prise en charge par IBM MQ. Cela signifie que, dans certains cas, un caractère précomposé dans le CCSID source sera converti en une séquence de caractères de combinaison dans le CCSID cible, ou l'inverse.

**Remarque :** IBM MQ ne prenant pas en charge les CCSID du gestionnaire de files d'attente UTF-16 , les données d'en-tête de message ne peuvent pas être codées en UTF-16.

## Prise en charge d'Unicode par IBM MQ AIX



Sous IBM MQ for AIX , la conversion vers et depuis les CCSID Unicode pris en charge (de préférence 1200 ou 1208) est prise en charge pour les CCSID non Unicode dans la liste suivante:

037  
273, 278, 280, 284, 285, 297  
423, 437  
500  
813, 819, 850, 852, 856, 857, 858, 860, 861, 865, 867, 869, 875, 878, 880  
901, 902, 912, 915, 916, 920, 923, 924, 932, 933, 935, 937, 938, 939, 942, 943, 948, 949, 950, 954, 964, 970  
1026, 1046, 1089  
1129, 1130, 1131, 1132, 1133, 1140, 1141, 1142, 1143, 1144, 1145, 1146, 1147, 1148, 1149, 1153, 1156, 1157  
1200, 1208, 1250, 1251, 1253, 1254, 1258, 1280, 1281, 1282, 1283, 1284, 1285  
1363, 1364, 1381, 1383, 1386, 1388  
4899  
5026, 5035, 5050, 5346, 5347, 5348, 5349, 5350, 5351, 5352, 5353, 5354, 5488  
9044, 9048, 9449  
12712  
13488  
17584  
33722

## Prise en charge d'Unicode par IBM MQ for Windows et Linux



Sous IBM MQ for Windows et IBM MQ for Linux vers et depuis les CCSID Unicode pris en charge (de préférence 1200 ou 1208) sont prises en charge pour les CCSID non Unicode de la liste suivante:

037,  
277, 278, 280, 284, 285, 290, 297  
300, 301  
420, 424, 437  
500  
813, 819, 833, 835, 836, 837, 838, 850, 852, 855, 856, 857, 858, 860, 861, 862, 863, 864, 865, 866, 867, 868, 869, 870, 871, 874, 875, 878, 880, 891, 897  
901, 902, 903, 904, 912, 913«5», à la page 1037, 915, 916, 918, 920, 921, 922, 923, 924, 927, 928, 930, 931«1», à la page 1037, 932«2», à la page 1037, 933, 935, 937, 938«3», à la page 1037, 939, 941, 942, 943, 947, 948, 949, 950, 951, 954«4», à la page 1037, 964, 970  
1006, 1025, 1026, 1027, 1040, 1041, 1042, 1043, 1046, 1047, 1051, 1088, 1089, 1097, 1098

1112, 1114, 1115, 1122, 1123, 1124, 1129, 1130, 1132, 1133, 1140, 1141, 1142, 1143, 1144, 1145, 1146, 1147, 1148, 1149, 1153, 1156, 1157  
1200, 1208, 1250, 1251, 1252, 1253, 1254, 1255, 1256, 1257, 1258, 1275, 1280, 1281, 1282, 1283  
1363, 1364, 1374, 1375, 1376, 1377, 1378, 1379, 1380, 1381, 1383, 1386, 1388  
4899  
5050, 5346, 5347, 5348, 5349, 5350, 5351, 5352, 5353, 5354, 5488«5», à la page 1037  
9044, 9048, 9449  
12712  
13488  
17584  
33722«4», à la page 1037

#### Remarques :

1. 931 utilise 939 pour la conversion.
2. 932 utilise 942 pour la conversion.
3. 938 utilise 948 pour la conversion.
4. 954 et 33722 utilisent 5050 pour la conversion.
5. Sous Windows et Linux uniquement.

### Prise en charge d'Unicode par IBM i



Pour plus de détails sur la prise en charge d'UNICODE, reportez-vous à la publication IBM i appropriée relative à votre système d'exploitation.

### Prise en charge d'Unicode par IBM MQ for z/OS



Sous IBM MQ for z/OS , la conversion vers et depuis les CCSID Unicode pris en charge (de préférence 1200 ou 1208) est prise en charge pour les CCSID non Unicode dans la liste suivante:

37  
256, 259, 273, 275, 277, 278, 280, 282, 284, 285, 290, 293, 297  
300, 301, 367  
420, 423, 424, 437  
500  
720, 737, 775  
803, 806, 808, 813, 819, 833, 834, 835, 836, 837, 838, 848, 849, 850, 851, 852, 855, 856, 857, 858, 859, 860, 861, 862, 863, 864, 865, 866, 867, 868, 869, 870, 871, 872, 874, 875, 878, 880, 891, 895, 896, 897  
901, 902, 903, 904, 905, 912, 914, 915, 916, 918, 920, 921, 922, 923, 924, 927, 928, 930, 932, 933, 935, 937, 939, 941, 942, 943, 944, 946, 947, 948, 949, 950, 951  
1004, 1006, 1008, 1009, 1010, 1011, 1012, 1013, 1014, 1015, 1016, 1017, 1018, 1019, 1025, 1026, 1027, 1040, 1041, 1042, 1043, 1046, 1047, 1051, 1088, 1089, 1097, 1098  
1112, 1114, 1115, 1122, 1123, 1124, 1125, 1126, 1129, 1130, 1131, 1132, 1133, 1137, 1140, 1141, 1142, 1143, 1144, 1145, 1146, 1147, 1148, 1149, 1153, 1154, 1155, 1156, 1157, 1158, 1159, 1160, 1161, 1162, 1164  
1200, 1208, 1250, 1251, 1252, 1253, 1254, 1255, 1256, 1257, 1258, 1275, 1276, 1277, 1280, 1281, 1282, 1283, 1284, 1285  
1351, 1362, 1363, 1364, 1370, 1371, 1380, 1381, 1385, 1386, 1388, 1390, 1399  
4899, 4909, 4930, 4933, 4948, 4951, 4952, 4960, 4971  
5012 5039 5104 5123 5142 5210 5346 5347 5348 5349 5350 5351 5352 5353 5354 5488

8482 8612  
9027 9030 9044 9048 9049 9056 9061 9066 9238 9449  
1166  
12712  
13121, 13218, 13488, 1374, 1375, 1376, 1377, 1378, 1379  
16684, 16804  
17248, 17584  
21427  
28709

## Normes de codage sur les plateformes 64 bits

Utilisez ces informations pour en savoir plus sur les normes de codage sur les plateformes 64 bits et les types de données préférés.

### Types de données préférés

Ces types ne changent jamais de taille et sont disponibles sur les plateformes IBM MQ 32 bits et 64 bits:

Tableau 675. Noms et longueurs des types de données

Nom	Longueur
MQLONG	4 octets
MQULONG	4 octets
MQINT32	4 octets
MQUINT32	4 octets
MQINT64	8 octets
MQUINT64	8 octets

## Types de données standard sous AIX, Linux, and Windows

Découvrez les types de données standard sur les applications AIX and Linux32 bits et AIX, Linux, and Windows 64 bits.

### Applications AIX and Linux 32 bits




Tableau 676. Noms et longueurs des types de données pour les applications AIX and Linux 32 bits

Nom	Longueur
char	1 octet
short	2 octets
int	4 octets
long	4 octets
float	4 octets
double	8 octets
long double	8 octets
pointeur	4 octets
ptrdiff_t	4 octets

Tableau 676. Noms et longueurs des types de données pour les applications AIX and Linux 32 bits (suite)

Nom	Longueur
taille_t	4 octets
time_t	4 octets
clock_t	4 octets
wchar_t	4 octets


 Notez que sous AIX , wchar\_t a une valeur de 2 octets.

## Applications AIX and Linux 64 bits



Tableau 677. Noms et longueurs des types de données pour les applications AIX and Linux 64 bits

Nom	Longueur
char	1 octet
short	2 octets
int	4 octets
long	8 octets
float	4 octets
double	8 octets
long double	8 octets
pointeur	8 octets
ptrdiff_t	8 octets
taille_t	8 octets
time_t	8 octets
clock_t	4 octets
wchar_t	4 octets

 Notez que sous AIX , wchar\_t a une valeur de 2 octets.

## Applications Windows 64 bits



Tableau 678. Noms et longueurs des types de données pour les applications Windows 64 bits

Nom	Longueur
char	1 octet
short	2 octets
int	4 octets
long	4 octets
float	4 octets
double	8 octets

Tableau 678. Noms et longueurs des types de données pour les applications Windows 64 bits (suite)

Nom	Longueur
long double	8 octets
pointeur	8 octets
	Notez que tous les pointeurs sont de 8 octets.
ptrdiff_t	8 octets
taille_t	8 octets
time_t	8 octets
clock_t	4 octets
wchar_t	2 octets
WORD	2 octets
DWORD	4 octets
HANDLE	8 octets
HFILE	4 octets

## Remarques sur le codage sous Windows

### Windows

#### DESCRIPTEUR hf ;

Utilisation

```
hf = CreateFile((LPCTSTR) FileName,
                Access,
                ShareMode,
                xihSecAttsNTRestrict,
                Create,
                AttrAndFlags,
                NULL);
```

Ne pas utiliser

```
HFILE hf;
hf = (HFILE) CreateFile((LPCTSTR) FileName,
                        Access,
                        ShareMode,
                        xihSecAttsNTRestrict,
                        Create,
                        AttrAndFlags,
                        NULL);
```

car cela génère une erreur.

#### size\_t len fgets

Utilisation

```
size_t len
while (fgets(string1, (int) len, fp) != NULL)
len = strlen(buffer);
```

Ne pas utiliser

```
int len;
```



```
while (fgets(string1, len, fp) != NULL)
len = strlen(buffer);
```

## printf

### Utilisation

```
printf("My struc pointer: %p", pMyStruc);
```

### Ne pas utiliser

```
printf("My struc pointer: %x", pMyStruc);
```

Si vous avez besoin d'une sortie hexadécimale, vous devez imprimer les 4 octets supérieurs et inférieurs séparément.

## char \* ptr

### Utilisation

```
char * ptr1;
char * ptr2;
size_t bufLen;

bufLen = ptr2 - ptr1;
```

### Ne pas utiliser

```
char *ptr1;
char *ptr2;
UINT32 bufLen;

bufLen = ptr2 - ptr1;
```

## alignBytes

### Utilisation

```
alignBytes = (unsigned short) ((size_t) address % 16);
```

### Ne pas utiliser

```
void *address;
unsigned short alignBytes;

alignBytes = (unsigned short) ((UINT32) address % 16);
```

## len

### Utilisation

```
len = (UINT32) ((char *) address2 - (char *) address1);
```

### Ne pas utiliser

```
void *address1;
void *address2;
UINT32 len;

len = (UINT32) ((char *) address2 - (char *) address1);
```

## sscanf

### Utilisation

```
MQLONG SBCSprt;  
sscanf(line, "%d", &SBCSprt);
```

### Ne pas utiliser

```
MQLONG SBCSprt;  
sscanf(line, "%1d", &SBCSprt);
```

%1d tente d'insérer un type à 8 octets dans un type à 4 octets ; n'utilisez %1 que si vous utilisez un type de données long réel. MQLONG, UINT32 et INT32 sont définis comme étant quatre octets, identiques à un int sur toutes les plateformes IBM MQ :

## IBM i IBM i Application Programming Reference (ILE/RPG)

Programmation d'application pour IBM i.

Utilisez ces informations pour vous aider à développer des applications pour IBM i.

- [«Description des types de données sous IBM i», à la page 1043](#)
- [«Appels de fonction sur IBM i», à la page 1311](#)
- [«Attributs des objets sur IBM i», à la page 1435](#)
- [«Applications», à la page 1484](#)
- [«Codes retour pour IBM i \(ILE RPG\)», à la page 1498](#)
- [«Règles de validation des options MQI pour IBM i \(ILE RPG\)», à la page 1499](#)
- [«Codages de machine sur IBM i», à la page 1502](#)
- [«Options de rapport et indicateurs de message sous IBM i», à la page 1505](#)

## Dépréciation du mode compatibilité pour les applications RPG et COBOL sous IBM i

### IBM i

Depuis IBM MQ for IBM i 9.0, le produit ne prend plus en charge les applications RPG ou COBOL qui utilisent la liaison dynamique appelée mode compatibilité. Ce mode de fonctionnement était nécessaire pour les applications écrites avant MQSeries 5.1, et les versions ultérieures du produit fournissaient un environnement d'exécution compatible pour ces applications, même si les fichiers de stockage nécessaires à leur compilation n'étaient pas présents dans IBM WebSphere MQ 6.0. La liaison dynamique (mode compatibilité) était fournie par les programmes suivants dans la bibliothèque QMQM, qui ne sont plus disponibles dans IBM MQ for IBM i 9.0 :

- AMQVSTUB
- AMQZSTUB
- QMQM
- MQCLOSE
- MQCONN
- MQDISC
- MQGET
- MQINQ
- MQOPEN
- MQPUT

- MQPUT1
- MQSET

Depuis la IBM MQ for IBM i 9.0, les applications qui utilisent ce mode compatibilité pour fonctionner doivent être recompilées afin d'utiliser les appels MQ liés statiques qui sont mis à disposition par les programmes de service LIBMQM et LIBMQM\_R. Des exemples de programme, tels qu'AMQ3PUT4 et AMQ3GET4, vous montrent comment utiliser ce modèle de programmation. Pour plus d'informations sur l'utilisation de ces appels MQ, voir [Référence de programmation des applications IBM i \(ILE RPG\)](#).

#### Remarques :

- Vous devez coder à nouveau les applications, qui utilisent actuellement l'interface CALL 'QMQM', pour qu'elles utilisent le programme de service LIBMQM à la place.

Les objets de programme et les programmes de service de la liste précédente, par exemple, QMQM, MQCONN, MQPUT, AMQVSTUB et AMQZSTUB, sont supprimés dans IBM MQ for IBM i 9.0 et les applications codées pour utiliser le mode compatibilité cessent de fonctionner.

- Si des applications sont liées au programme de service LIBMQM à l'adresse IBM MQ for IBM i 8.0, il n'est pas nécessaire de recompiler ou de relier ces applications à l'adresse IBM MQ for IBM i 9.0 ou ultérieure.
- Il n'est pas possible d'installer plusieurs versions de IBM MQ for IBM i sur la même partition.

Pour déterminer si votre programme RPG ou COBOL utilise le mode compatibilité, utilisez la commande **DSPPGMREF** (Display Program References) pour afficher les programmes externes appelés par le programme d'application. S'il existe des références aux programmes répertoriés dans cette section, le programme ne s'exécutera pas dans IBM MQ for IBM i 9.0 ou une version ultérieure. L'exemple suivant de sortie de la commande **DSPPGMREF** présente trois objets de programme dépréciés, MQCONN, MQOPEN et MQCLOSE :

```

Program . . . . . : MYAPPPGM
Library . . . . . : MYLIB
Text 'description'. . . . . : ILE/COBOL SAMPLE PUT TO QUEUE (MQPUT)
Number of objects referenced . . . . . : 5
Object . . . . . : MQCONN
Library . . . . . : *LIBL
Object type . . . . . : *PGM
Object . . . . . : MQOPEN
Library . . . . . : *LIBL
Object type . . . . . : *PGM
Object . . . . . : MQCLOSE
Library . . . . . : *LIBL
Object type . . . . . : *PGM

```

Ces programmes doivent être recompilés à l'aide de la méthode d'appel de procédure lié (Bound Procedural Call) décrite dans [Preparing COBOL programs in IBM i](#).

Si vous tentez d'exécuter un programme d'application à IBM MQ for IBM i 9.0 ou une version ultérieure qui utilise le mode compatibilité, la première erreur la plus courante est une erreur MCH3401 qui tente d'appeler le programme MQCONN ou QMQM.

#### Tâches associées

[Développement d'applications](#)

## IBM i Description des types de données sous IBM i

Cette collection de rubriques fournit des descriptions des types de données utilisés dans la programmation IBM i.

### Conventions utilisées dans la description des types de données

Pour chaque type de données élémentaires, ces informations donnent une description de son utilisation, sous une forme indépendante du langage de programmation. Cette opération est suivie de déclarations standard dans la version ILE du langage de programmation RPG. Les définitions des types de données élémentaires sont incluses ici pour assurer la cohérence. RPG utilise des spécifications 'D' dans lesquelles

les zones de travail peuvent être déclarées à l'aide des attributs dont vous avez besoin. Toutefois, vous pouvez le faire dans les spécifications de calcul où la zone est utilisée.

Pour utiliser les types de données élémentaires, vous devez créer:

- Un membre /COPY contenant tous les types de données, ou
- Une structure de données externe (PF) contenant tous les types de données. Vous devez ensuite spécifier vos zones de travail avec les attributs 'LIKE' comme zone de type de données appropriée.

Les avantages de la deuxième option sont que les définitions peuvent être utilisées en tant que 'FIELD REFERENCE FILE' pour d'autres objets IBM i. Si une définition de type de données IBM MQ change, il est relativement simple de recréer ces objets.

## Types de données élémentaires

Tous les autres types de données décrits dans cette section correspondent soit directement à ces types de données élémentaires, soit à des agrégats de ces types de données élémentaires (tableaux ou structures).

<i>Tableau 679. Types de données élémentaires</i>	
<b>Type de données</b>	<b>Représentation</b>
MQBOOL	Entier signé à 10 chiffres
MQBYTE	Zone alphanumérique à 1 octet
MQBYTE16	Zone alphanumérique à 16 octets
MQBYTE24	Zone alphanumérique de 24 octets
MQBYTE32	Zone alphanumérique de 32 octets
MQBYTE64	Zone alphanumérique à 64 octets
MQCHAR	Zone alphanumérique à 1 octet
MQCHAR4	Zone alphanumérique à 4 octets
MQCHAR8	Zone alphanumérique à 8 octets
MQCHAR12	Zone alphanumérique à 12 octets
MQCHAR16	Zone alphanumérique à 16 octets
MQCHAR20	Zone alphanumérique de 20 octets
MQCHAR28	Zone alphanumérique de 28 octets
MQCHAR32	Zone alphanumérique de 32 octets
MQCHAR48	Zone alphanumérique de 48 octets
MQCHAR64	Zone alphanumérique à 64 octets
MQCHAR128	Zone alphanumérique de 128 octets
MQCHAR256	Zone alphanumérique de 256 octets
MQFLOAT32	Nombre à virgule flottante sur 4 octets
MQFLOAT64	Nombre à virgule flottante sur 8 octets
MQHCONFIG	Descripteur de configuration
MQHCONN	Entier signé à 10 chiffres
MQHMSG	Descripteur de message qui donne accès à un message

Tableau 679. Types de données élémentaires (suite)

Type de données	Représentation
MQHOBJ	Entier signé à 10 chiffres
MQINT8	Entier signé 8 bits
MQINT16	Entier signé 16 bits
MQINT32	Entier signé 32 bits
MQINT64	Entier signé sur 64 bits
MQLONG	Entier signé 32 bits
MQPID	Identificateur de processus
MQPTR	Pointeur
MQTID	Identificateur d'unité d'exécution
MQUINT8	Entier non signé 8 bits
MQUINT16	Entier non signé 16 bits
MQUINT32	entier non marqué 32-bit
MQUINT64	Entier non signé sur 64 bits
MQULONG	entier non marqué 32-bit
PMQACH	Pointeur vers une structure de données de type MQACH
PMQAIR	Pointeur vers une structure de données de type MQAIR
PMQAXC	Pointeur vers une structure de données de type MQAXC
PMAXP	Pointeur vers une structure de données de type MAXP
PMQBMHO	Pointeur vers une structure de données de type MQBMHO
PMQBO	Pointeur vers une structure de données de type MQBO
PMQBOOL	Pointeur vers des données de type MQBOOL
PMQBYTE	Pointeur vers des données de type MQBYTE
PMQBYTE <sub>n</sub>	Pointeur vers des données de type MQBYTE <sub>n</sub>
PMQCBC	Pointeur vers une structure de données de type MQCBC
PMQCBD	Pointeur vers une structure de données de type MQCBD
PMQCHAR	Pointeur vers une structure de données de type MQCHAR
PMQCHARV	Pointeur vers une structure de données de type MQCHARV
PMQCHAR <sub>n</sub>	Pointeur vers des données de type MQCHAR <sub>n</sub>

Tableau 679. Types de données élémentaires (suite)

Type de données	Représentation
PMQCIH	Pointeur vers une structure de données de type MQCIH
PMQCMHO	Pointeur vers une structure de données de type MQCMHO
PMQCNO	Pointeur vers une structure de données de type MQCNO
PMQCSP	Pointeur vers une structure de données de type MQCSP
PMQCTLO	Pointeur vers une structure de données de type MQCTLO
PMQDH	Pointeur vers une structure de données de type MQDH
PMQDHO	Pointeur vers une structure de données de type MQDHO
PMQDLH	Pointeur vers une structure de données de type MQDLH
PMQDMHO	Pointeur vers une structure de données de type MQDMHO
PMQDMPO	Pointeur vers une structure de données de type MQDMPO
PMQEPH	Pointeur vers une structure de données de type MQEPH
PMQFLOAT32	Pointeur vers des données de type MQFLOAT32
PMQFLOAT64	Pointeur vers des données de type MQFLOAT64
PMQFUNC	Pointeur vers une fonction
PMQGMO	Pointeur vers une structure de données de type MQGMO
PMQHCONFIG	Pointeur vers des données de type MQHCONFIG
PMQHCONN	Pointeur vers des données de type MQHCONN
PMQHMSG	Pointeur vers des données de type MQHMSG
PMQHOBJ	Pointeur vers des données de type MQHOBJ
PMQIIH	Pointeur vers une structure de données de type MQIIH
PMQIMPO	Pointeur vers une structure de données de type MQIMPO
PMQINT8	Pointeur vers des données de type MQINT8
PMQINT16	Pointeur vers des données de type MQINT16
PMQINT32	Pointeur vers des données de type MQINT32
PMQINT64	Pointeur vers des données de type MQINT64

Tableau 679. Types de données élémentaires (suite)

Type de données	Représentation
PMQLONG	Pointeur vers des données de type MQLONG
PMQMD	Pointeur vers une structure de données de type MQMD
PMQMDE	Pointeur vers une structure de données de type MQMDE
PMQMD1	Pointeur vers une structure de données de type MQMD1
PMQMD2	Pointeur vers une structure de données de type MQMD2
PMQMHBO	Pointeur vers une structure de données de type MQMHBO
PMQOD	Pointeur vers une structure de données de type MQOD
PMQOR	Pointeur vers une structure de données de type MQOR
PMQPD	Pointeur vers une structure de données de type MQPD
PMQPID	Pointeur vers un identificateur de processus MQPID
PMQPMO	Pointeur vers une structure de données de type MQPMO
PMQPTR	Pointeur vers des données de type MQPTR
PMQRFH	Pointeur vers une structure de données de type MQRFH
PMQRFH2	Pointeur vers une structure de données de type MQRFH2
PMQRMH	Pointeur vers une structure de données de type MQRMH
PMQRR	Pointeur vers une structure de données de type MQRR
PMQSCO	Pointeur vers une structure de données de type MQSCO
PMQSD	Pointeur vers une structure de données de type MQSD
PMQSMPO	Pointeur vers une structure de données de type MQSMPO
PMQSRO	Pointeur vers une structure de données de type MQSRO
PMQSTS	Pointeur vers une structure de données de type MQSTS
PMQTID	Pointeur vers un identificateur d'unité d'exécution MQTID

Tableau 679. Types de données élémentaires (suite)

Type de données	Représentation
PMQTM	Pointeur vers une structure de données de type MQTM
PMQTM2	Pointeur vers une structure de données de type MQTM2
PMQUINT8	Pointeur vers les données de type MQUINT8
PMQUINT16	Pointeur vers des données de type MQUINT16
PMQUINT32	Pointeur vers des données de type MQUINT32
PMQUINT64	Pointeur vers les données de type MQUINT64
PPMQULONG	Pointeur vers les données de type MQULONG
ID PMQVOID	Pointeur
PMQWIH	Pointeur vers une structure de données de type MQWIH
PMQXQH	Pointeur vers une structure de données de type MQXQH

### IBM i MQBOOL sur IBM i

Le type de données MQBOOL représente une valeur booléenne. La valeur 0 représente false. Toute autre valeur représente true.

Un MQBOOL doit être aligné comme pour le type de données MQLONG.

### IBM i MQBYTE sur IBM i

Le type de données MQBYTE représente un octet de données unique.

Aucune interprétation particulière n'est placée sur l'octet-il est traité comme une chaîne de bits, et non comme un nombre binaire ou un caractère. Aucun alignement spécial n'est requis.

Un tableau de MQBYTE est parfois utilisé pour représenter une zone de mémoire principale dont la nature est inconnue du gestionnaire de files d'attente. Par exemple, la zone peut contenir des données de message d'application ou une structure. L'alignement des limites de cette zone doit être compatible avec la nature des données qu'elle contient.

### IBM i MQBYTEn (chaîne de n octets) sur IBM i

Chaque type de données MQBYTEn représente une chaîne de  $n$  octets.

Où  $n$  peut prendre l'une des valeurs suivantes:

- 16, 24, 32 ou 64.

Chaque octet est décrit par le type de données MQBYTE. Aucun alignement spécial n'est requis.

Si les données de la chaîne sont plus courtes que la longueur définie de la chaîne, les données doivent être complétées par des valeurs nulles pour remplir la chaîne.

Lorsque le gestionnaire de files d'attente renvoie des chaînes d'octets à l'application (par exemple, sur l'appel MQGET), le gestionnaire de files d'attente remplit toujours avec des valeurs nulles à la longueur définie de la chaîne.

Des constantes sont disponibles pour définir les longueurs des zones de chaîne d'octets.



### **IBM i** **MQCHAR (caractère) sous IBM i**

Le type de données MQCHAR représente un caractère unique.

L'identificateur de jeu de caractères codés du caractère est celui du gestionnaire de files d'attente (voir l'attribut **CodedCharSetId** dans la rubrique [CodedCharSetId](#) ). Aucun alignement spécial n'est requis.

**Remarque :** Les données de message d'application spécifiées dans les appels MQGET, MQPUT et MQPUT1 sont décrites par le type de données MQBYTE et non par le type de données MQCHAR.

### **IBM i** **MQCHARn (chaîne de n caractères) sur IBM i**

Chaque type de données MQCHARn représente une chaîne de *n* caractères.

Où *n* peut prendre l'une des valeurs suivantes:

- 4, 8, 12, 16, 20, 28, 32, 48, 64, 128 ou 256

Chaque caractère est décrit par le type de données MQCHAR. Aucun alignement spécial n'est requis.

Si les données de la chaîne sont plus courtes que la longueur définie de la chaîne, les données doivent être complétées par des blancs pour remplir la chaîne. Dans certains cas, un caractère nul peut être utilisé pour terminer la chaîne prématurément, au lieu d'être rempli avec des blancs ; le caractère nul et les caractères qui le suivent sont traités comme des blancs, jusqu'à la longueur définie de la chaîne. Les emplacements où une valeur nulle peut être utilisée sont identifiés dans les descriptions d'appel et de type de données.

Lorsque le gestionnaire de files d'attente renvoie des chaînes de caractères à l'application (par exemple, sur l'appel MQGET), le gestionnaire de files d'attente remplit toujours avec des blancs la longueur définie de la chaîne ; le gestionnaire de files d'attente n'utilise pas le caractère nul pour délimiter la chaîne.

Des constantes sont disponibles pour définir les longueurs des zones de chaîne de caractères.

### **IBM i** **MQFLOAT32 sur IBM i**

Le type de données MQFLOAT32 est un nombre à virgule flottante 32 bits représenté à l'aide du format à virgule flottante IEEE standard.

Un MQFLOAT32 doit être aligné sur une limite de 4 octets.

### **IBM i** **MQFLOAT64 sur IBM i**

Le type de données MQFLOAT64 est un nombre à virgule flottante 64 bits représenté à l'aide du format à virgule flottante IEEE standard.

Un MQFLOAT64 doit être aligné sur une limite de 8 octets.

### **MQHCONFIG-descripteur de configuration**

Le type de données MQHCONFIG représente un descripteur de configuration, c'est-à-dire le composant configuré pour un service installable particulier. Un descripteur de configuration doit être aligné sur sa limite naturelle.

**Remarque :** Les applications doivent tester les variables de ce type uniquement pour l'égalité.

### **IBM i** **MQHCONN (descripteur de connexion) sous IBM i**

Le type de données MQHCONN représente un descripteur de connexion, c'est-à-dire la connexion à un gestionnaire de files d'attente particulier.

Un descripteur de connexion doit être aligné sur sa limite naturelle.

**Remarque :** Les applications doivent tester les variables de ce type uniquement pour l'égalité.

### **IBM i** **MQHMSG (descripteur de message) sous IBM i**

Le type de données MQHMSG représente un descripteur de message qui donne accès à un message.

Un descripteur de message doit être aligné sur une limite de 8 octets.

**Remarque :** Les applications doivent tester les variables de ce type uniquement pour l'égalité.

#### **IBM i MQHOBJ (descripteur d'objet) sur IBM i**

Le type de données MQHOBJ représente un descripteur d'objet qui donne accès à un objet.

Un descripteur d'objet doit être aligné sur sa limite naturelle.

**Remarque :** Les applications doivent tester les variables de ce type uniquement pour l'égalité.

#### **IBM i MQINT8 (entier signé de 8 bits) sur IBM i**

Le type de données MQINT8 est un entier signé de 8 bits qui peut prendre n'importe quelle valeur comprise entre -128 et +127, sauf indication contraire du contexte.

#### **IBM i MQINT16 (entier signé 16 bits) sur IBM i**

Le type de données MQINT16 est un entier signé 16 bits qui peut prendre n'importe quelle valeur comprise entre -32 768 et +32 767, sauf indication contraire du contexte.

Un MQINT16 doit être aligné sur une limite de 2 octets.

#### **IBM i MQINT32 (entier 32 bits) sous IBM i**

Le type de données MQINT32 est un entier signé 32 bits.

Elle est équivalente à MQLONG.

#### **IBM i MQINT64 (entier 64 bits) sous IBM i**

Le type de données MQINT64 est un entier signé de 64 bits qui peut prendre n'importe quelle valeur comprise entre -9 223 372 036 854 775 808 et + 9 223 372 036 854 775 807, sauf restriction par le contexte.

Pour COBOL, la plage valide est limitée à -999 999 999 999 999 999 999 à +999 999 999 999 999 999. Un MQINT64 doit être aligné sur une limite de 8 octets.

#### **IBM i MQLONG (entier long) sous IBM i**

Le type de données MQLONG est un entier binaire signé de 32 bits qui peut prendre n'importe quelle valeur comprise entre -2 147 483 648 et + 2 147 483 647, sauf restriction par le contexte, aligné sur sa limite naturelle.

### **MQPID-identificateur de processus**

Identificateur de processus IBM MQ .

Il s'agit du même identificateur utilisé dans les vidages IBM MQ trace et FFST , mais il peut être différent de l'identificateur de processus du système d'exploitation.

### **MQPTR-pointeur**

Le type de données MQPTR est l'adresse des données de n'importe quel type. Un pointeur doit être aligné sur sa limite naturelle ; il s'agit d'une limite de 16 octets sur IBM i.

Certains langages de programmation prennent en charge les pointeurs typés ; l'interface MQI les utilise également dans quelques cas.

### **MQTID-Identificateur d'unité d'exécution**

Identificateur de l'unité d'exécution MQ .

Il s'agit du même identificateur utilisé dans la trace MQ et les vidages FFST , mais il peut être différent de l'identificateur d'unité d'exécution du système d'exploitation.

### **IBM i MQUINT8 (entier non signé de 8 bits) sur IBM i**

Le type de données MQUINT8 est un entier non signé de 8 bits qui peut prendre n'importe quelle valeur comprise entre 0 et +255, sauf indication contraire du contexte.

### **MQUINT16 -Entier non signé 16 bits**

Le type de données MQUINT16 est un entier non signé de 16 bits qui peut prendre n'importe quelle valeur comprise entre 0 et +65 535, sauf restriction par le contexte.

Une MQUINT16 doit être alignée sur une limite de 2 octets.

### **IBM i MQUINT32 (entier non signé 32 bits) sur IBM i**

Le type de données MQUINT32 est un entier non signé 32 bits. Il est équivalent à MQULONG.

### **MQUINT64 -Entier non signé 64 bits**

Le type de données MQUINT64 est un entier non signé de 64 bits qui peut prendre n'importe quelle valeur comprise entre 0 et +18 446 744 073 709 551 615, sauf restriction par le contexte.

Pour COBOL, la plage valide est comprise entre 0 et +999 999 999 999 999 999. Un MQUINT64 doit être aligné sur une limite de 8 octets.

### **MQULONG-Entier non signé 32 bits**

Le type de données MQULONG est un entier binaire non signé de 32 bits qui peut prendre n'importe quelle valeur comprise entre 0 et + 4 294 967 294, sauf restriction par le contexte.

Une valeur MQULONG doit être alignée sur une limite de 4 octets.

### **PMQACH-pointeur vers une structure de données de type MQACH**

Pointeur vers une structure de données de type MQACH.

### **PMQAIR-pointeur vers une structure de données de type MQAIR**

Pointeur vers une structure de données de type MQAIR.

### **PMQAXC-pointeur vers une structure de données de type MQAXC**

Pointeur vers une structure de données de type MQAXC.

### **PMQAXP-pointeur vers une structure de données de type MQAXP**

Pointeur vers une structure de données de type MQAXP.

### **PMQBMHO-pointeur vers une structure de données de type MQBMHO**

Pointeur vers une structure de données de type MQBMHO.

### **PMQBO-pointeur vers une structure de données de type MQBO**

Pointeur vers une structure de données de type MQBO.

***PMQBOOL-pointeur vers des données de type MQBOOL***

Pointeur vers des données de type MQBOOL.

Pointeur vers des données de type MQBOOL.

***PMQBYTE-pointeur vers un type de données de MQBYTE***

Un pointeur vers un type de données de MQBYTE.

***PMQBYTEn-pointeur vers une structure de données de type MQBYTEn***

Un pointeur vers une structure de données de type MQBYTEn, où n peut être 8, 12, 16, 24, 32, 40, 48 ou 128.

***PMQCBC-pointeur vers une structure de données de type MQCBC***

Pointeur vers une structure de données de type MQCBC.

***PMQCBD-pointeur vers une structure de données de type MQCBD***

Pointeur vers une structure de données de type MQCBD.

***PMQCHAR-pointeur vers des données de type MQCHAR***

Pointeur vers des données de type MQCHAR.

***PMQCHARV-pointeur vers une structure de données de type MQCHARV***

Pointeur vers une structure de données de type MQCHARV.

***PMQCHARn-pointeur vers un type de données de MQCHARn***

Un pointeur vers un type de données de MQCHARn, où n peut être 4, 8, 12, 20, 28, 32, 64, 128, 256, 264.

***PMQCIH-pointeur vers une structure de données de type MQCIH***

Pointeur vers une structure de données de type MQCIH.

***PMQCMHO-pointeur vers une structure de données de type MQCMHO***

Pointeur vers une structure de données de type MQCMHO.

***PMQCNO-pointeur vers une structure de données de type MQCNO***

Pointeur vers une structure de données de type MQCNO.

***PMQCSP-pointeur vers une structure de données de type MQCSP***

Pointeur vers une structure de données de type MQCSP.

***PMQCTLO-pointeur vers une structure de données de type MQCTLO***

Pointeur vers une structure de données de type MQCTLO.

***PMQDH-pointeur vers une structure de données de type MQDH***

Pointeur vers une structure de données de type MQDH.

***PMQDHO-pointeur vers une structure de données de type MQDHO***

Pointeur vers une structure de données de type MQDHO.

***PMQDLH-pointeur vers une structure de données de type MQDLH***

Pointeur vers une structure de données de type MQDLH.

***PMQDMHO-pointeur vers une structure de données de type MQDMHO***

Pointeur vers une structure de données de type MQDMHO.

***PMQDMPO-pointeur vers une structure de données de type MQDMPO***

Pointeur vers une structure de données de type MQDMPO.

Pointeur vers une structure de données de type MQDMPO.

***PMQEPPH-pointeur vers une structure de données de type MQEPPH***

Pointeur vers une structure de données de type MQEPPH.

***PMQFLOAT32 -Pointeur vers les données de type MQFLOAT32***

Un pointeur vers des données de type MQFLOAT32.

***PMQFLOAT64 -pointeur vers des données de type MQFLOAT64***

Un pointeur vers des données de type MQFLOAT64.

***PMQFUNC-pointeur vers une fonction***

Un pointeur vers une fonction.

***PMQGMO-pointeur vers une structure de données de type MQGMO***

Pointeur vers une structure de données de type MQGMO.

***PMQHCONFIG-pointeur vers un type de données de MQHCONFIG***

Un pointeur vers un type de données de MQHCONFIG.

***PMQHCONN-pointeur vers un type de données de MQHCONN***

Un pointeur vers un type de données de MQHCONN.

***PMQHMSG-pointeur vers un type de données de MQHMSG***

Un pointeur vers un type de données de MQHMSG.

***PMQHOBJ-pointeur vers des données de type MQHOBJ***

Pointeur vers des données de type MQSMPO.

***PMQIIH-pointeur vers une structure de données de type MQIIH***

Pointeur vers une structure de données de type MQIIH.

***PMQIMPO-pointeur vers une structure de données de type MQIMPO***

Pointeur vers une structure de données de type MQIMPO.

***PMQINT8 -Pointeur vers des données de type MQINT8***

Un pointeur vers des données de type MQINT8.

***PMQINT16 -Pointeur vers des données de type MQINT16***

Un pointeur vers des données de type MQINT16.

***IBM i PMQINT32 (Pointeur vers des données de type MQINT32) sous IBM i***

Le type de données PMQINT32 est un pointeur vers des données de type MQINT32. Elle est équivalente à PMQLONG.

***IBM i PMQINT64 (Pointeur vers des données de type MQINT64) sous IBM i***

Le type de données PMQINT64 est un pointeur vers des données de type MQINT64.

***PMQLONG-pointeur vers des données de type MQLONG***

Pointeur vers des données de type MQLONG.

***PMQMD-pointeur vers la structure de type MQMD***

Pointeur vers la structure de type MQMD.

***PMQMDE-pointeur vers une structure de données de type MQMDE***

Pointeur vers une structure de données de type MQMDE.

***PMQMDI-pointeur vers une structure de données de type MQMDI***

Pointeur vers une structure de données de type MQMDI.

***PMQMD2 -pointeur vers une structure de données de type MQMD2***

Un pointeur vers une structure de données de type MQMD2

***PMQMHBO-pointeur vers une structure de données de type MQMHBO***

Pointeur vers une structure de données de type MQMHBO.

***PMQOD-pointeur vers une structure de données de type MQOD***

Pointeur vers une structure de données de type MQOD.

***PMQOR-pointeur vers une structure de données de type MQOR***

Pointeur vers une structure de données de type MQOR.

***PMQPD-pointeur vers une structure de données de type MQPD***

Pointeur vers une structure de données de type MQPD.

***PMQPID-pointeur vers un identificateur de processus***

Pointeur vers un identificateur de processus.

***PMQPMO-pointeur vers une structure de données de type MQPMO***

Pointeur vers une structure de données de type MQPMO.

***PMQPTR-pointeur vers des données de type MQPTR***

Pointeur vers des données de type MQPTR.

***PMQRFH-pointeur vers une structure de données de type MQRFH***

Pointeur vers une structure de données de type MQRFH.

***PMQRFH2 -pointeur vers une structure de données de type MQRFH2***

Un pointeur vers une structure de données de type MQRFH2.

***PMQRMH-pointeur vers une structure de données de type MQRMH***

Pointeur vers une structure de données de type MQRMH.

***PMQRR-pointeur vers une structure de données de type MQRR***

Pointeur vers une structure de données de type MQRR.

***PMQSCO-pointeur vers une structure de données de type MQSCO***

Pointeur vers une structure de données de type MQSCO.

***PMQSD-pointeur vers une structure de données de type MQSD***

Pointeur vers une structure de données de type MQSD.

***PMQSMPO-pointeur vers une structure de données de type MQSMPO***

Pointeur vers une structure de données de type MQSMPO.

***PMQSRO-pointeur vers une structure de données de type MQSRO***

Pointeur vers une structure de données de type MQSRO.

***PMQSTS-pointeur vers une structure de données de type MQSTS***

Pointeur vers une structure de données de type MQSTS.

***PMQTID-pointeur vers une structure de données de type MQTID***

Pointeur vers une structure de données de type MQTID.

***PMQTM-pointeur vers une structure de données de type MQTM***

Pointeur vers une structure de données de type MQTM.

***PMQTM2 -pointeur vers une structure de données de type MQTM2***

Un pointeur vers une structure de données de type MQTM2.

***PMQUINT8 -pointeur vers des données de type MQUINT8***

Un pointeur vers des données de type MQUINT8.

***PMQUINT16 -pointeur vers des données de type MQUINT16***

Un pointeur vers des données de type MQUINT16.

***IBM i PMQUINT32 (pointeur vers des données de type MQUINT32) sur IBM i***

Le type de données PMQUINT32 est un pointeur vers des données de type MQUINT32. Elle est équivalente à PMQULONG.

***IBM i PMQUINT64 (Pointeur vers des données de type MQUINT64) sur IBM i***

Le type de données PMQUINT64 est un pointeur vers des données de type MQUINT64.

***PMQULONG-pointeur vers les données de type MQULONG***

Un pointeur vers des données de type MQULONG.



## ***PMQVOID-pointeur***

Un pointeur.

## ***PMQWIH-pointeur vers une structure de données de type MQWIH***

Pointeur vers une structure de données de type MQWIH.

## ***PMQXQH-pointeur vers une structure de données de type MQXQH***

Pointeur vers une structure de données de type MQXQH.

## **Considérations linguistiques**

Cette rubrique contient des informations pour vous aider à utiliser l'interface MQI du langage de programmation RPG.

Voici quelques remarques relatives à la langue:

- «Copier des fichiers», à la page [1057](#)
- «Appels», à la page [1059](#)
- «Paramètres d'appel», à la page [1059](#)
- «Structures», à la page [1060](#)
- «Constante nommée», à la page [1060](#)
- «Procédures MQI», à la page [1060](#)
- «Remarques sur les unités d'exécution», à la page [1061](#)
- «Commitment Control», à la page [1061](#)
- «Codage des appels liés», à la page [1061](#)
- «Conventions de notation», à la page [1062](#)

## **Copier des fichiers**

Divers fichiers COPY sont fournis pour faciliter l'écriture des programmes d'application RPG qui utilisent la mise en file d'attente de messages. Il existe trois ensembles de fichiers COPY:

- Les fichiers COPY dont le nom se termine par la lettre *G* sont destinés aux programmes qui utilisent des liaisons statiques. Ces fichiers sont initialisés avec les exceptions indiquées dans «Structures», à la page [1060](#).
- Les fichiers COPY dont le nom se termine par la lettre *H* sont destinés à être utilisés avec des programmes qui utilisent des liens statiques, mais qui ne sont **pas** initialisés.
- Les fichiers COPY dont le nom se termine par la lettre *R* sont destinés à être utilisés avec des programmes qui utilisent des liens dynamiques. Ces fichiers sont initialisés avec les exceptions indiquées dans «Structures», à la page [1060](#).

Les fichiers COPY se trouvent dans QRPGLSRC dans la bibliothèque QMQM.

Pour chaque ensemble de fichiers COPY, il existe deux fichiers contenant des constantes nommées et un fichier pour chacune des structures. Les fichiers COPY sont récapitulés dans [Tableau 680, à la page 1058](#).

Tableau 680. Fichiers RPG COPY

Nom de fichier (liaison statique, initialisée, CMQ* G)	Nom de fichier (liaison statique, non initialisée, CMQ* H)	Nom de fichier (liaison dynamique, initialisée, CMQ* R)	Contenu
CMQBOG	CMQBOH	-	Structure des options de début
CMQCDG	CMQCDH	CMQCDR	Structure de définition de canal
CMQCFBFG	CMQCFBFH	-	Paramètre de filtre de bits PCF
CMQCFG	-	-	Constantes pour PCF et événements
CMQCFBSG	CMQCFBSH	-	Chaîne d'octets PCF
CMQCFGRG	CMQCFGRH	-	Paramètre de groupe PCF
CMQCFOP	CMQCFIFH	-	Paramètre de filtre d'entier PCF
CMQCFHG	CMQCFHH	-	En-tête PCF
CMQCFILG	CMQCFILH	-	Structure des paramètres de liste d'entiers PCF
CMQCFING	CMQCFINH	-	Structure des paramètres de type entier PCF
CMQCFSG	CMQCFSFH	-	Paramètre de filtre de chaîne PCF
CMQCFSLG	CMQCFSLH	-	Structure des paramètres de liste de chaînes PCF
CMQCFSTG	CMQCFSTH	-	Structure des paramètres de chaîne PCF
CMQCFXLG	CMQCFXLH	-	Nom abrégé PCF pour CFIL64
CMQCFXNG	CMQCFXNH	-	Nom abrégé PCF pour CFIN64
CMQCIHG	CMQCIHH	-	Structure d'en-tête d'informations CICS
CMQCNOG	CMQCNOH	-	Structure des options de connexion
CMQCSPG	CMQCSPH	-	Paramètres de sécurité
CMQCXPG	CMQCXPH	CMQCXPR	Structure des paramètres d'exit de canal
CMQDHG	CMQDHH	CMQDHR	Structure d'en-tête de distribution
CMQDLHG	CMQDLHH	CMQDLHR	Structure d'en-tête de rebut
CMQDXPG	CMQDXPH	CMQDXPR	Structure des paramètres d'exit de conversion de données
CMQEPHG	CMQEPHH	-	Structure d'en-tête PCF intégrée
CMQG	-	CMQR	Constantes nommées pour l'interface MQI principale
CMQGMOG	CMQGMOH	CMQGMOR	Obtenir la structure des options de message
CMQIIHG	CMQIIHH	CMQIIHR	Structure d'en-tête d'informations IMS
CMQMDEG	CMQMDEH	CMQMDER	Structure d'extension du descripteur de message

Tableau 680. Fichiers RPG COPY (suite)

Nom de fichier (liaison statique, initialisée, CMQ* G)	Nom de fichier (liaison statique, non initialisée, CMQ* H)	Nom de fichier (liaison dynamique, initialisée, CMQ* R)	Contenu
CMQOMD	CMQMDH	CMQMDR	Structure de descripteur de message
CMQMD1G	CMQMD1H	CMQMD1R	Structure de descripteur de message version 1
CMQMD2G	CMQMD2H	-	Structure de descripteur de message version 2
CMQODG	CMQODH	CMQODR	Structure de descripteur d'objet
CMQORG	CMQORH	CMQORR	Structure d'enregistrement d'objet
CMQPMOG	CMQPMOH	CMQPMOR	Structure des options de message d'insertion
CMQPSG	-	-	Constantes pour la publication / l'abonnement
CMQRFHG	CMQRFHH	-	Règles et structure d'en-tête de formatage
CMQRFH2G	CMQRFH2H	-	Règles et structure de l'en-tête 2 de mise en forme
CMQRMHG	CMQRMHH	CMQRMHR	Structure d'en-tête de message de référence
CMQRRG	CMQRRH	CMQRRR	Structure d'enregistrement de réponse
CMQTMCG	CMQTMCH	CMQTMCR	Structure du message de déclenchement (format de caractères)
CMQTMCG	CMQTMCH	CMQTMCR	Structure du message de déclenchement (format de caractère) version 2
CMQTMG	CMQTMH	CMQTMR	Structure des messages de déclenchement
CMQWIHG	CMQWIHH	-	Structure d'en-tête des informations de travail
CMQXG	-	CMQXR	Constantes nommées pour l'exit de conversion de données
CMQXQHG	CMQXQHH	CMQXQHR	Structure d'en-tête de file d'attente de transmission

## Appels

Les appels sont décrits à l'aide de leurs noms individuels.

## Paramètres d'appel

Certains paramètres transmis à l'interface MQI peuvent avoir plusieurs fonctions simultanées. En effet, la valeur entière transmise est souvent testée sur la définition de bits individuels dans la zone, et non sur sa

valeur totale. Cela vous permet d'ajouter plusieurs fonctions ensemble et de les transmettre en tant que paramètre unique.

## Structures

Toutes les structures IBM MQ sont définies avec des valeurs initiales pour les zones, avec les exceptions suivantes:

- Toute structure dont le suffixe est H.
- MQTMC
- MQTMC2

Ces valeurs initiales sont définies dans le tableau correspondant à chaque structure.

Les déclarations de structure ne contiennent pas d'instructions DS . Cela permet à l'application de déclarer une structure de données unique ou une structure de données à occurrences multiples, en codant l'instruction DS , puis en utilisant l'instruction /COPY pour la copier dans le reste de la déclaration:

```
D*..1.....2.....3.....4.....5.....6.....7
D* Declare an MQMD data structure with 5 occurrences
DMYMD      DS              5
D/COPY CMQMDR
```

## Constante nommée

Il existe de nombreuses valeurs entières et de caractères qui fournissent un échange de données entre votre programme d'application et le gestionnaire de files d'attente. Pour faciliter une approche plus lisible et cohérente de l'utilisation de ces valeurs, des constantes nommées sont définies pour celles-ci. Vous pouvez utiliser ces constantes nommées et non les valeurs qu'elles représentent, car cela améliore la lisibilité du code source du programme.

Lorsque le fichier COPY CMQG est inclus dans un programme pour définir les constantes, le compilateur RPG émet de nombreux messages de gravité zéro pour les constantes qui ne sont pas utilisées par le programme ; ces messages sont bénins et peuvent être ignorés en toute sécurité.

## Procédures MQI

Lorsque vous utilisez les appels liés ILE, vous devez vous connecter aux procédures MQI lorsque vous créez votre programme. Ces procédures sont exportées à partir des programmes de service suivants, selon le cas:

### QMQM/LIBMQM

Ce programme de service contient les liaisons à unité d'exécution unique pour la version 5.1 et les versions ultérieures. Reportez-vous à la section suivante pour connaître les remarques particulières à prendre en compte lors de l'écriture d'applications à unités d'exécution.

### QMQM/LIBMQM\_R

Ce programme de service contient les liaisons à unités d'exécution multiples pour la version 5.1 et les versions ultérieures. Reportez-vous à la section suivante pour connaître les remarques particulières à prendre en compte lors de l'écriture d'applications à unités d'exécution.

### QMQM/LIBMQIC

Ce programme de service permet de lier des applications client sans unités d'exécution.

### QMQM/LIBMQIC\_R

Ce programme de service permet de lier des applications client à unités d'exécution.

Utilisez la commande CRTPGM pour créer vos programmes. Par exemple, la commande suivante crée un programme à unité d'exécution unique qui utilise les appels liés à ILE:

```
CRTPGM PGM(MYPROGRAM) BNDSRVPGM(QMQM/LIBMQM)
```

## Remarques sur les unités d'exécution

Le compilateur RPG utilisé pour IBM i fait partie de WebSphere Development Toolset et de WebSphere Development Studio for IBM i et est connu sous le nom de compilateur ILE RPG IV.

En général, les programmes RPG ne doivent pas utiliser les programmes de service à unités d'exécution multiples. Les exceptions sont les programmes RPG créés à l'aide du compilateur ILE RPG IV et contenant le mot clé THREAD (\*SERIALIZE) dans la spécification de contrôle. Cependant, même si ces programmes sont sûrs pour les unités d'exécution, une attention particulière doit être accordée à la conception globale de l'application, car THREAD (\*SERIALIZE) force la sérialisation des procédures RPG au niveau du module, ce qui peut avoir un impact négatif sur les performances globales.

Lorsque des programmes RPG sont utilisés comme exits de conversion de données, ils doivent être sécurisés par unité d'exécution et recompilés à l'aide du compilateur ILE RPG 4.4 ou supérieur, avec THREAD (\*SERIALIZE) indiqué dans la spécification de contrôle.

Pour plus d'informations sur l'utilisation d'unités d'exécution, voir *IBM i IBM MQ Development Studio: ILE RPG Reference* et *IBM i IBM MQ Development Studio: ILE RPG Programmer's Guide*.

## Commitment Control

Les fonctions de point de synchronisation MQI MQCMIT et MQBACK sont disponibles pour les programmes ILE RPG exécutés en mode normal ; ces appels permettent au programme de valider et d'éliminer les modifications apportées aux ressources MQ .

## Codage des appels liés

Les procédures MQI ILE sont répertoriées dans [Tableau 681](#), à la page 1061.

<i>Tableau 681. Appels liés ILE RPG pris en charge par chaque programme de service</i>		
<b>Nom de l'appel</b>	<b>LIBMQM et LIBMQM_R</b>	<b>LIBMQIC et LIBMQIC_R</b>
MQBACK	Y	Y
MQBEGIN	Y	Y
MQCMIT	Y	Y
MQCLOSE	Y	Y
MQCONN	Y	Y
MQCONNX	Y	Y
MQDISC	Y	Y
MQGET	Y	Y
MQINQ	Y	Y
MQOPEN	Y	Y
MQPUT	Y	Y
MQPUT1	Y	Y
MQSET	Y	Y
MQXCNVC	Y	Y

Pour utiliser ces procédures, vous devez:

1. Définissez les procédures externes dans vos spécifications'D'. Tous ces éléments sont disponibles dans le membre de fichier COPY CMQG contenant les constantes nommées.
2. Utilisez le code opération CALLP pour appeler la procédure avec ses paramètres.

Par exemple, l'appel MQOPEN requiert l'inclusion du code suivant:

```
D*****
D** MQOPEN Call -- Open Object (From COPY file CMQG) **
D*****
D*
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
DMQOPEN PR EXTPROC('MQOPEN')
D* Connection handle
D HCONN 10I 0 VALUE
D* Object descriptor
D OBJDSC 224A
D* Options that control the action of MQOPEN
D OPTS 10I 0 VALUE
D* Object handle
D HOBJ 10I 0
D* Completion code
D CMPCOD 10I 0
D* Reason code qualifying CMPCOD
D REASON 10I 0
D*
```

Pour appeler la procédure, après avoir initialisé les différents paramètres, vous avez besoin du code suivant:

```
...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8
C CALLP MQOPEN(HCONN : MQOD : OPTS : HOBJ :
C CMPCOD : REASON)
```

Ici, la structure MQOD est définie à l'aide du membre COPY CMQODG qui la décompose en ses composants.

## Conventions de notation

Les dernières rubriques de cette section montrent comment:

- Les appels doivent être appelés
- Les paramètres doivent être déclarés
- Différents types de données doivent être déclarés

Dans un certain nombre de cas, les paramètres sont des tableaux ou des chaînes de caractères dont la taille n'est pas fixe. Pour celles-ci, une minuscule "n" est utilisée pour représenter une constante numérique. Lorsque la déclaration de ce paramètre est codée, la valeur "n" doit être remplacée par la valeur numérique requise.



## MQAIR (enregistrement d'informations d'authentification) sur IBM i

La structure MQAIR représente l'enregistrement des informations d'authentification.

### Présentation

**Objectif:** La structure MQAIR permet à une application s'exécutant en tant que client IBM MQ de spécifier des informations sur un authentificateur à utiliser pour la connexion client. La structure est un paramètre d'entrée dans l'appel MQCONN.

**Jeu de caractères et codage:** les données de MQAIR doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par ENNAT.

- «Zones», à la page [1063](#)
- «Valeurs initiales», à la page [1065](#)
- «Déclaration RPG», à la page [1065](#)

## Zones

La structure MQAIR contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

### AICN (entier signé à 10 chiffres)

Il s'agit du nom d'hôte ou de l'adresse réseau d'un hôte sur lequel le serveur LDAP s'exécute. Il peut être suivi d'un numéro de port facultatif, entre parenthèses.

Si la valeur est inférieure à la longueur de la zone, terminez la valeur par un caractère NULL ou remplissez la zone avec des blancs. Si la valeur n'est pas valide, l'appel échoue avec le code anomalie RC2387.

Le numéro de port par défaut est 389.

Il s'agit d'une zone d'entrée. La longueur de cette zone est donnée par LNAICN. La valeur initiale de cette zone est des caractères blancs.

### AITYP (entier signé à 10 chiffres)

Il s'agit du type d'informations d'authentification contenues dans l'enregistrement.

La valeur doit être:

#### AITPLD

Révocation de certificat à l'aide du serveur LDAP.

Si la valeur n'est pas valide, l'appel échoue avec le code anomalie RC2386.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est AITLDP.

### AIPW (entier signé à 10 chiffres)

Il s'agit du mot de passe requis pour accéder au serveur CRL LDAP.

Si la valeur est inférieure à la longueur de la zone, terminez la valeur par un caractère NULL ou remplissez la zone avec des blancs. Si le serveur LDAP ne requiert pas de mot de passe ou si vous omettez le nom d'utilisateur LDAP, *AIPW* doit être null ou vide. Si vous omettez le nom d'utilisateur LDAP et que *AIPW* n'est pas null ou vide, l'appel échoue avec le code anomalie RC2390.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par LNLDPW. Valeur initiale des caractères blancs de cette zone.

### AILUL (entier signé à 10 chiffres)

Il s'agit de la longueur en octets du nom d'utilisateur LDAP indiqué par la zone *AILUP* ou *AILUO* . La valeur doit être comprise entre zéro et LNDISN. Si la valeur n'est pas valide, l'appel échoue avec le code anomalie RC2389.

Si le serveur LDAP impliqué ne requiert pas de nom d'utilisateur, définissez cette zone sur zéro.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

### AILUO (entier signé à 10 chiffres)

Il s'agit du décalage en octets du nom d'utilisateur LDAP à partir du début de la structure MQAIR.

Le décalage peut être positif ou négatif. La zone est ignorée si *LDAPUserNameLength* a la valeur zéro.

Vous pouvez utiliser *LDAPUserNamePtr* ou *LDAPUserNameOffset* pour spécifier le nom d'utilisateur LDAP, mais pas les deux ; voir la description de la zone *LDAPUserNamePtr* pour plus de détails.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

### AILUP (entier signé à 10 chiffres)

Il s'agit du nom d'utilisateur LDAP.

Il se compose du nom distinctif de l'utilisateur qui tente d'accéder au serveur CRL LDAP. Si la valeur est inférieure à la longueur spécifiée par *AILUL*, terminez la valeur par un caractère null ou remplissez la valeur avec des blancs pour la longueur *AILUL*. La zone est ignorée si *AILUL* a la valeur zéro.

Vous pouvez fournir le nom d'utilisateur LDAP de l'une des deux manières suivantes:

- A l'aide de la zone de pointeur *AILUP*

Dans ce cas, l'application peut déclarer une chaîne distincte de la structure MQAIR et définir *AILUP* sur l'adresse de la chaîne.

Envisagez d'utiliser *AILUP* pour les langages de programmation qui prennent en charge le type de données de pointeur d'une manière portable dans différents environnements (par exemple, le langage de programmation C).

- En utilisant la zone de décalage *AILUO*

Dans ce cas, l'application doit déclarer une structure composée contenant la structure MQSCO suivie du tableau d'enregistrements MQAIR suivi des chaînes de nom d'utilisateur LDAP et définir *AILUO* sur le décalage de la chaîne de nom appropriée à partir du début de la structure MQAIR. Vérifiez que cette valeur est correcte et qu'elle peut être prise en compte dans un MQLONG (le langage de programmation le plus restrictif est COBOL, pour lequel la plage valide est comprise entre -999 999 999 et +999 999 999).

Envisagez d'utiliser *AILUO* pour les langages de programmation qui ne prennent pas en charge le type de données de pointeur ou qui implémentent le type de données de pointeur d'une manière qui peut ne pas être portable dans des environnements différents (par exemple, le langage de programmation COBOL).

Quelle que soit la technique choisie, n'utilisez qu'une seule des méthodes *AILUP* et *AILUO* ; l'appel échoue avec le code anomalie RC2388.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire.

**Remarque :** Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

### **AISID (entier signé à 10 chiffres)**

La valeur doit être:

#### **AISIDV**

Identificateur de l'enregistrement d'informations d'authentification.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est AISIDV.

### **AIVER (entier signé à 10 chiffres)**

La valeur doit être:

#### **AIVER1**

Enregistrement des informations d'authentification Version-1 .

La constante suivante indique le numéro de version de la version en cours:

#### **AIVERC**

Version actuelle de l'enregistrement des informations d'authentification.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est AIVER1.



## Valeurs initiales

Tableau 682. Zones dans MQAIR pour MQAIR		
Nom de zone	Nom de la constante	Valeur de la constante
AISID	AISIDV	'AIR~'
AIVER	IversC	1
AITYP	AITPLD	1
AICN	Aucun	Chaîne nulle ou blancs
AILUP	Aucun	Pointeur null ou octets null
AILUO	Aucun	0
AILUL	Aucun	0
AIPW	Aucun	Chaîne nulle ou blancs

### Remarques :

1. Le symbole ~ représente un caractère blanc unique.

## Déclaration RPG

```
D*.1.....2.....3.....4.....5.....6.....7..
D* MQAIR Structure
D*
D* Structure identifier
D AISID 1 4 INZ('AIR ')
D* Structure version number
D AIVER 5 8I 0 INZ(1)
D* Type of authentication information
D AITYP 9 12I 0 INZ(1)
D* Connection name of CRL LDAP server
D AICN 13 276 INZ
D* Address of LDAP user name
D AILUP 277 292* INZ(*NULL)
D* Offset of LDAP user name from start of MQAIR structure
D AILUO 293 296I 0 INZ(0)
D* Length of LDAP user name
D AILUL 297 300I 0 INZ(0)
D* Password to access LDAP server
D AIPW 301 332 INZ
```

## MQBMHO (Buffer to message handle options) sur IBM i

Structure définissant les options de mémoire tampon pour le traitement des messages.

### Présentation

**Objectif:** La structure MQBMHO permet aux applications de spécifier des options qui contrôlent la façon dont les descripteurs de message sont générés à partir des mémoires tampon. La structure est un paramètre d'entrée dans l'appel MQBUFMH.

**Jeu de caractères et codage:** les données dans MQBMHO doivent être dans le jeu de caractères de l'application et le codage de l'application (ENNAT).

- «Zones», à la page 1066
- «Valeurs initiales», à la page 1066
- «Déclaration RPG», à la page 1066

## Zones

La structure MQBMHO contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

### **BMSID (entier signé à 10 chiffres)**

Mémoire tampon pour la structure de descripteur de message-Zone StrucId .

Il s'agit de l'identificateur de structure. La valeur doit être:

#### **BBMSIDV**

Identificateur de la mémoire tampon pour la structure de descripteur de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est BMSIDV.

### **BMVER (entier signé à 10 chiffres)**

Zone Buffer to message handle structure-Version.

Il s'agit du numéro de version de la structure. La valeur doit être:

#### **BMVER1**

Numéro de version de la mémoire tampon pour la structure de descripteur de message.

La constante suivante indique le numéro de version de la version en cours:

#### **BBMVERVC**

Version en cours de la structure de mémoire tampon à descripteur de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est BMVER1.

### **BMOPT (entier signé à 10 chiffres)**

Structure de la mémoire tampon vers le descripteur de message-Zone Options.

La valeur peut être :

#### **BMDLPR**

Les propriétés ajoutées au descripteur de message sont supprimées de la mémoire tampon. Si l'appel échoue, aucune propriété n'est supprimée.

Options par défaut: si vous n'avez pas besoin de l'option décrite, utilisez l'option suivante:

#### **BBMAUCUN**

Aucune option spécifiée.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est BMDLPR.

## Valeurs initiales

Tableau 683. Zones dans MQBMHO		
Nom de zone	Nom de la constante	Valeur de la constante
<i>BMSID</i>	BBMSIDV	'BMHO'
<i>BMVER</i>	BMVER1	1
<i>BMOPT</i>	BBMAUCUN	0

## Déclaration RPG

```
D* MQBMHO Structure
D*
D*
D* Structure identifier
D BMSID          1      4    INZ('BMHO')
D*
D* Structure version number
D BMVER          5      8I 0 INZ(1)
D*
```

```
D* Options that control the action of MQBUFMH
D BMOPT          9      12I 0 INZ(1)
```

## IBM i MQBO (options de début) sous IBM i

La structure MQBO permet à l'application de spécifier des options relatives à la création d'une unité de travail.

### Présentation

**Objectif:** la structure est un paramètre d'entrée-sortie dans l'appel MQBEGIN.

**Jeu de caractères et codage:** les données dans MQBO doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par ENNAT.

- «Zones», à la page [1067](#)
- «Valeurs initiales», à la page [1068](#)
- «Déclaration RPG», à la page [1068](#)

### Zones

La structure MQBO contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

#### **BOOPT (entier signé à 10 chiffres)**

Options qui contrôlent l'action de MQBEGIN.

La valeur doit être:

##### **BONONE**

Aucune option spécifiée.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est BONONE.

#### **BOSID (chaîne de caractères de 4 octets)**

Identificateur de structure.

La valeur doit être:

##### **BOSIDV**

Identificateur de la structure des options de début.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est BOSIDV.

#### **BOVER (entier signé à 10 chiffres)**

Numéro de version de la structure.

La valeur doit être:

##### **BOVER1**

Numéro de version de la structure des options de début.

La constante suivante indique le numéro de version de la version en cours:

##### **BOVERC**

Version actuelle de la structure des options de début.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est BOVER1.

## Valeurs initiales

Tableau 684. Zones dans MQBO		
Nom de zone	Nom de la constante	Valeur de la constante
BOSID	BOSIDV	'BO- -'
BOVER	BOVER1	1
BOOPT	BONONE	0

### Remarques :

1. Le symbole - représente un caractère blanc unique.

## Déclaration RPG

```
D*..1.....2.....3.....4.....5.....6.....7..
D* MQBO Structure
D*
D* Structure identifieur
D BOSID 1 4 INZ('BO ')
D* Structure version number
D BOVER 5 8I 0 INZ(1)
D* Options that control the action of MQBEGIN
D BOOPT 9 12I 0 INZ(0)
```

## IBM i MQCBC (contexte de rappel) sur IBM i

Structure décrivant la routine de rappel.

### Présentation

#### Objet

La structure MQCBC permet de spécifier les informations de contexte qui sont transmises à une fonction de rappel.

La structure est un paramètre d'entrée-sortie dans l'appel à une routine de consommateur de message.

#### Version

La version actuelle de MQCBC est CBCV2.

#### Jeu de caractères et codage

Les données de MQCBC se trouvent dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par ENNAT. Toutefois, si l'application s'exécute en tant que client IBM MQ, la structure se trouve dans le jeu de caractères et le codage du client.

- [«Zones»](#), à la page 1068
- [«Valeurs initiales»](#), à la page 1074
- [«Déclaration RPG»](#), à la page 1074

### Zones

La structure MQCBC contient les zones suivantes ; les zones sont décrites par ordre alphabétique:

#### CBCBUFFLEN (entier signé à 10 chiffres)

La mémoire tampon peut être supérieure à la valeur MaxMsgLength définie pour le consommateur et à la valeur ReturnedLength dans le MQGMO.

Structure de contexte de rappel-Zone BufferLength .

Il s'agit de la longueur en octets de la mémoire tampon de messages qui a été transmise à cette fonction.

La longueur réelle des messages est indiquée dans la zone DataLength .

L'application peut utiliser la totalité de la mémoire tampon à ses propres fins pendant la durée de la fonction de rappel.

Il s'agit d'une zone d'entrée de la fonction de consommateur de message ; elle n'est pas pertinente pour une fonction de gestionnaire d'exceptions.

### **CBCCALLBA (entier signé à 10 chiffres)**

Structure de contexte de rappel-Zone CallbackArea .

Il s'agit d'une zone disponible que la fonction de rappel peut utiliser.

Le gestionnaire de files d'attente ne prend aucune décision en fonction du contenu de cette zone et est transmis tel quel à partir de la zone CBDCALLBA de la structure MQCBD, qui est un paramètre de l'appel MQCB utilisé pour définir la fonction de rappel.

Les modifications apportées à *CBCCALLBA* sont conservées dans les appels de la fonction de rappel pour un *CBCHOBJ*. Cette zone n'est pas partagée avec les fonctions de rappel pour les autres descripteurs.

Il s'agit d'un champ d'entrée/sortie de la fonction de rappel. La valeur initiale de cette zone est un pointeur NULL ou des octets NULL.

### **CBCCALLT (entier signé à 10 chiffres)**

Structure de contexte de rappel-Zone CallType .

Zone contenant des informations sur la raison pour laquelle cette fonction a été appelée. Les types d'appel suivants sont définis.

Types d'appel de distribution de message: ces types d'appel contiennent des informations sur un message. Les paramètres **CBCCLEN** et **CBCCBUFFLEN** sont valides pour ces types d'appel.

#### **RMC**

La fonction de consommateur de message a été appelée avec un message qui a été supprimé de façon destructive de l'identificateur d'objet.

Si la valeur de *CBCCC* est CCWARN, la valeur de la zone *Reason* est RC2079 ou l'un des codes indiquant un problème de conversion de données.

#### **Réseau CBCTMN**

La fonction de consommateur de message a été appelée avec un message qui n'a pas encore été supprimé de façon destructive de l'identificateur d'objet. Le message peut être supprimé de façon destructive de l'identificateur d'objet à l'aide de *MsgToken*.

Le message n'a peut-être pas été supprimé pour les raisons suivantes:

- Les options MQGMO ont demandé une opération de navigation, GMBR\*
- Le message est plus grand que la mémoire tampon disponible et les options MQGMO ne spécifient pas *gmatm*

Si la valeur de *CBCCC* est CCWARN, la valeur de la zone *Reason* est RC2080 ou l'un des codes indiquant un problème de conversion de données.

Types d'appel de contrôle de rappel: ces types d'appel contiennent des informations sur le contrôle du rappel et ne contiennent pas de détails sur un message. Ces types d'appel sont demandés à l'aide de CBDOPT dans la structure MQCBD.

Les paramètres **CBCCLEN** et **CBCCBUFFLEN** ne sont pas valides pour ces types d'appel.

#### **CBCTRC**

Le but de ce type d'appel est de permettre à la fonction de rappel d'effectuer une configuration initiale.

La fonction de rappel est appelée immédiatement après l'enregistrement du rappel, c'est-à-dire lors du retour d'un appel MQCB à l'aide d'une valeur pour la zone *Operation* de CBREG.

Ce type d'appel est utilisé à la fois pour les consommateurs de messages et les gestionnaires d'événements.

S'il est demandé, il s'agit du premier appel de la fonction de rappel.

La valeur de la zone *CBCREA* est RCNONE.

### **CCTSC**

Le but de ce type d'appel est de permettre à la fonction de rappel d'effectuer une configuration lors de son démarrage, par exemple, en réinstanciant des ressources qui ont été nettoyées lors de son arrêt précédent.

La fonction de rappel est appelée lorsque la connexion est démarrée à l'aide de CTLSR ou de CTLSW.

Si une fonction de rappel est enregistrée dans une autre fonction de rappel, ce type d'appel est appelé lorsque le rappel est renvoyé.

Ce type d'appel est utilisé uniquement pour les consommateurs de message.

La valeur de la zone *CBCREA* est RCNONE.

### **CCTTC**

Le but de ce type d'appel est de permettre à la fonction de rappel d'effectuer un nettoyage lorsqu'elle est arrêtée pendant un certain temps, par exemple, en nettoyant des ressources supplémentaires qui ont été acquises lors de la consommation de messages.

La fonction de rappel est appelée lorsqu'un appel MQCTL est émis à l'aide d'une valeur pour la zone *Operation* de CTLSP.

Ce type d'appel est utilisé uniquement pour les consommateurs de message.

La valeur de la zone *CBCREA* est définie pour indiquer la raison de l'arrêt.

### **CCTDC**

Le but de ce type d'appel est de permettre à la fonction de rappel d'effectuer un nettoyage final à la fin du processus de consommation. La fonction de rappel est appelée lorsque:

- La fonction de rappel est désenregistrée à l'aide d'un appel MQCB avec BCUNR.
- La file d'attente est fermée, ce qui entraîne un désenregistrement implicite. Dans cette instance, la fonction de rappel est transmise à HOUNUH en tant que descripteur d'objet.
- L'appel MQDISC se termine-provoquant une fermeture implicite et, par conséquent, un désenregistrement. Dans ce cas, la connexion n'est pas immédiatement déconnectée et toute transaction en cours n'est pas encore validée.

Si l'une de ces actions est effectuée dans la fonction de rappel elle-même, l'action est appelée une fois que le rappel est renvoyé.

Ce type d'appel est utilisé à la fois pour les consommateurs de messages et les gestionnaires d'événements.

S'il est demandé, il s'agit du dernier appel de la fonction de rappel.

La valeur de la zone *CBCREA* est définie pour indiquer la raison de l'arrêt.

### **CBCTEC**

#### **Fonction du gestionnaire d'événements**

La fonction de gestionnaire d'événements a été appelée sans message lorsque:

- Un appel MQCTL est émis avec une valeur pour la zone *Operation* de CTLSP, ou
- Le gestionnaire de files d'attente ou la connexion s'arrête ou se met au repos.

Cet appel peut être utilisé pour effectuer l'action appropriée pour toutes les fonctions de rappel.

### • Fonction de consommateur de message

La fonction de consommateur de message a été appelée sans message lorsqu'une erreur (CBCCC = CCFAIL) spécifique au descripteur d'objet a été détectée ; par exemple, CBCREA code = RC2016 .

La valeur de la zone CBCREA est définie pour indiquer la raison de l'appel.

Il s'agit d'une zone d'entrée. CBCTMR et CMCTMN ne s'appliquent qu'aux fonctions de consommateur de messages.

### **CBCCC (entier signé à 10 chiffres)**

Structure de contexte de rappel-Zone CompCode .

Il s'agit du code achèvement. Il indique s'il y a eu des problèmes lors de la consommation du message. Il s'agit de l'un des éléments suivants:

#### **CCOK**

Achèvement réussi

#### **CCWARN**

Avertissement (achèvement partiel)

#### **CCFAIL**

Echec de l'appel

Il s'agit d'une zone d'entrée. La valeur initiale de ce champ est CCOK.

### **CBCCONNAREA (entier signé à 10 chiffres)**

Structure de contexte de rappel-Zone ConnectionArea .

Il s'agit d'une zone disponible que la fonction de rappel peut utiliser.

Le gestionnaire de files d'attente ne prend aucune décision en fonction du contenu de cette zone et est transmis tel quel à partir de la zone ConnectionArea de la structure MQCTLO, qui est un paramètre de l'appel MQCTL utilisé pour contrôler la fonction de rappel.

Toutes les modifications apportées à cette zone par les fonctions de rappel sont conservées dans les appels de la fonction de rappel. Cette zone peut être utilisée pour transmettre des informations qui doivent être partagées par toutes les fonctions de rappel. Contrairement à *CallbackArea*, cette zone est commune à tous les rappels d'un descripteur de connexion.

Il s'agit d'une zone d'entrée et de sortie. La valeur initiale de cette zone est un pointeur NULL ou des octets NULL.

### **CBCLLEN (entier signé à 10 chiffres)**

Il s'agit de la longueur en octets des données d'application du message. Si la valeur est zéro, cela signifie que le message ne contient pas de données d'application.

La zone CBCLLEN contient la longueur du message, mais pas nécessairement la longueur des données de message transmises au destinataire. Il se peut que le message ait été tronqué. Utilisez la zone GMRL dans le MQGMO pour déterminer la quantité de données qui a été transmise au consommateur.

Si le code raison indique que le message a été tronqué, vous pouvez utiliser la zone CBCLLEN pour déterminer la taille réelle du message. Cela vous permet de déterminer la taille de la mémoire tampon requise pour accueillir les données de message, puis d'émettre un appel MQCB pour mettre à jour CBDMML dans le MQCBD avec une valeur appropriée.

Si l'option GMCONV est spécifiée, le message converti peut être supérieur à la valeur renvoyée pour DataLength. Dans de tels cas, l'application doit probablement émettre un appel MQCB pour mettre à jour le CBDMML dans le MQCBD afin qu'il soit supérieur à la valeur renvoyée par le gestionnaire de files d'attente pour DataLength.

Pour éviter les problèmes de troncature de message, spécifiez la longueur MaxMsgcomme CBDFM. Ainsi, le gestionnaire de files d'attente alloue une mémoire tampon pour la longueur complète

des messages après la conversion des données. Sachez toutefois que même si cette option est spécifiée, il est possible que l'espace de stockage disponible soit insuffisant pour traiter correctement la demande. Les applications doivent toujours vérifier le code anomalie renvoyé. Par exemple, s'il n'est pas possible d'allouer suffisamment de mémoire pour convertir le message, les messages sont renvoyés à l'application non convertie.

Il s'agit d'une zone d'entrée de la fonction de consommateur de message ; elle n'est pas pertinente pour une fonction de gestionnaire d'événements.

### **CBCFLG (entier signé à 10 chiffres)**

Indicateurs contenant des informations sur ce consommateur.

L'option suivante est définie:

#### **CBCFBE**

Cet indicateur peut être renvoyé si un appel MQCLOSE précédent utilisant l'option COQSC a échoué avec le code anomalie RC2458.

Ce code indique que le dernier message de lecture anticipée est renvoyé et que la mémoire tampon est à présent vide. Si l'application émet un autre appel MQCLOSE à l'aide de l'option COQSC, elle aboutit.

Notez qu'il n'est pas garanti qu'une application reçoit un message avec cet indicateur défini, car il peut encore y avoir des messages dans la mémoire tampon de lecture anticipée qui ne correspondent pas aux critères de sélection en cours. Dans cette instance, la fonction de consommateur est appelée avec le code anomalie RC2019 .

Si la mémoire tampon de lecture anticipée est vide, le consommateur est appelé avec l'indicateur CBCFBE et le code anomalie RC2518.

Il s'agit d'une zone d'entrée de la fonction de consommateur de message ; elle n'est pas pertinente pour une fonction de gestionnaire d'événements.

### **CBCHOBJ (entier signé à 10 chiffres)**

Structure de contexte de rappel-Zone CBCHOBJ.

Pour un appel à un consommateur de message, il s'agit du descripteur de l'objet relatif au consommateur de message.

Pour un gestionnaire d'événements, cette valeur est HONONE

L'application peut utiliser ce descripteur et le jeton de message dans le bloc Options d'extraction de message pour extraire le message si un message n'a pas été supprimé de la file d'attente.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est HOUNUH

### **CBRCRD (entier signé à 10 chiffres)**

**CBRCRD** indique la durée pendant laquelle le gestionnaire de files d'attente attend avant de tenter de se reconnecter. La zone peut être modifiée par un gestionnaire d'événements pour modifier complètement le délai ou arrêter la reconnexion.

Utilisez la zone **CBRCRD** uniquement si la valeur de la zone **Reason** dans le contexte de rappel est RC2545.

Lors de l'entrée dans le gestionnaire d'événements, la valeur de **CBRCRD** correspond au nombre de millisecondes pendant lesquelles le gestionnaire de files d'attente va attendre avant d'effectuer une tentative de reconnexion. Le [Tableau 685](#), à la [page 1072](#) répertorie les valeurs que vous pouvez définir pour modifier le comportement du gestionnaire de files d'attente en cas de retour du gestionnaire d'événements.

<i>Tableau 685. <b>CBRCRD</b> valeurs</i>	
<b>Valeur</b>	<b>Description</b>
-1	Ne faites plus de tentatives de reconnexion. Une erreur est renvoyée à l'application.



<i>Tableau 685. CBCRCD valeurs (suite)</i>	
Valeur	Description
0	Essayez de vous reconnecter immédiatement.
>0	Attendez ce nombre de millisecondes avant de retenter la connexion.

### **CBCREA (entier signé à 10 chiffres)**

Structure de contexte de rappel-Zone Raison.

Il s'agit du code anomalie qualifiant le CBCCC

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est RCNONE.

### **CBCSTATE (entier signé à 10 chiffres)**

Indication de l'état du consommateur en cours. Cette zone est la plus importante pour une application lorsqu'un code raison différent de zéro est transmis à la fonction de consommateur.

Vous pouvez utiliser cette zone pour simplifier la programmation d'application car vous n'avez pas besoin de coder le comportement de chaque code anomalie.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est CSNONE

<i>Tableau 686. Valeurs CBCSTATE et actions résultantes</i>		
Etat	Action du gestionnaire de files d'attente	Valeur de la constante
<i>CSNONE</i> Ce code anomalie représente un appel normal sans informations supplémentaires sur la raison	Non ; il s'agit du fonctionnement normal.	0
<i>CSSUST</i> Ces codes anomalie représentent des conditions temporaires.	La routine de rappel est appelée pour signaler la condition, puis suspendue. Au bout d'un certain temps, le système peut tenter à nouveau l'opération, ce qui peut entraîner une nouvelle levée de la même condition.	1
<i>CSSUSU</i> Ces codes anomalie représentent les conditions dans lesquelles le rappel doit agir pour résoudre la condition.	Le consommateur est suspendu et la routine de rappel est appelée pour signaler la condition. La routine de rappel doit résoudre la condition si possible et soit RESUME, soit fermer la connexion.	2
<i>CSSUS</i> Ces codes anomalie représentent des échecs qui empêchent d'autres rappels de message.	Le gestionnaire de files d'attente interrompt automatiquement la fonction de rappel. Si la fonction de rappel est reprise, il est probable qu'elle reçoive à nouveau le même code anomalie.	3
<i>CSSTOP</i> Ces codes anomalie représentent la fin de la consommation des messages.	Distribué au gestionnaire d'exceptions et aux rappels qui ont spécifié CBDTC. Aucun autre message ne peut être consommé.	4

### **CBCSID (entier signé à 10 chiffres)**

Structure de contexte de rappel-Zone StrucId .

Il s'agit de l'identificateur de structure ; la valeur doit être:

### **CBCSI**

Identificateur de la structure de contexte de rappel.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est CBCSI.

### **CBCVER (entier signé à 10 chiffres)**

Structure de contexte de rappel-Zone Version.

Il s'agit du numéro de version de la structure ; la valeur doit être:

### **CBCV1**

Structure de contexte de rappel Version-1 .

La constante suivante indique le numéro de version de la version en cours:

### **CBCCV**

Version actuelle de la structure de contexte de rappel.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est CBCV1.

## **Valeurs initiales**

<i>Tableau 687. Zones dans MQCBC</i>		
<b>Nom de zone</b>	<b>Nom de la constante</b>	<b>Valeur de la constante</b>
<i>CBCSID</i>	CBCSI	'CBC↵'
<i>CBCVER</i>	CBCV1	1
<i>CBCCALLT</i>	Aucun	0
<i>CBCHOBJ</i>	HOUNUH	-1
<i>CBCCALLBA</i>	Aucun	Pointeur null ou octets null
<i>CBCCONNAREA</i>	Aucun	Pointeur null ou octets null
<i>CBCCC</i>	CCOK	0
<i>CBCREA</i>	RCNONE	0
<i>CBCSTATE</i>	CSNONE (Aucun)	0
<i>CBCLLEN</i>	Aucun	0
<i>CBCBUFFLEN</i>	Aucun	0
<i>CBCFLG</i>	Aucun	0
<i>CBCRCD</i>	Aucun	0

### **Remarque :**

1. Le symbole ↵ représente un caractère blanc unique.

## **Déclaration RPG**

```
D* MQCBC Structure
D*
D*
D* Structure identifier
D  CBCSID           1      4   INZ('CBC ')
D*
D* Structure version number
D  CBCVER           5      8I 0 INZ(1)
```

```

D*
D* Why Function was called
D  CBCCALLT          9      12I 0 INZ(0)
D*
D* Object Handle
D  CBCHOBJ           13      16I 0 INZ(-1)
D*
D* Callback data passed to the function
D  CBCCALLBA        17      32*  INZ(*NULL)
D*
D* MQCTL Data area passed to the function
D  CBCCONNAREA     33      48*  INZ(*NULL)
D*
D* Completion Code
D  CBCCC           49      52I 0 INZ(0)
D*
D* Reason Code
D  CBCREA          53      56I 0 INZ(0)
D*
D* Consumer State
D  CBCSTATE        57      60I 0 INZ(0)
D*
D* Message Data Length
D  CBCLEN          61      64I 0 INZ(0)
D*
D* Buffer Length
D  CBCBUFFLEN     65      68I 0 INZ(0)
D*
** Flags containing information about
D* this consumer
D  CBCFLG          69      72I 0 INZ(0)
D* Ver:1 **
D* Number of milliseconds before reconnect attempt
D  CBCRCD          73      76I 0 INZ(0)
D* Ver:2 **
D*

```

## IBM i MQCBD (descripteur de rappel) sous IBM i

Structure spécifiant la fonction de rappel.

### Présentation

**Objectif:** La structure MQCBD est utilisée pour spécifier une fonction de rappel et les options contrôlant son utilisation par le gestionnaire de files d'attente.

La structure est un paramètre d'entrée dans l'appel MQCB.

**Version:** la version actuelle de MQCBD est CBDV1.

**Jeu de caractères et codage:** les données dans MQCBD doivent être dans le jeu de caractères et le codage du gestionnaire de files d'attente local ; elles sont fournies par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et ENNAT. Toutefois, si l'application s'exécute en tant que IBM MQ MQI client, la structure doit être dans le jeu de caractères et le codage du client.

- «Zones», à la page 1075
- «Valeurs initiales», à la page 1079
- «Déclaration RPG», à la page 1080

### Zones

La structure MQCBD contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

#### **CBDCALLBA (entier signé à 10 chiffres)**

Il s'agit d'une zone disponible que la fonction de rappel peut utiliser.

Le gestionnaire de files d'attente ne prend aucune décision en fonction du contenu de cette zone et est transmis sans modification à partir de la zone CBCCALLBA de la structure MQCBD, qui est un paramètre de la déclaration de fonction de rappel.

La valeur est utilisée uniquement sur un *Operation* ayant une valeur CBREG, sans rappel actuellement défini, elle ne remplace pas une définition précédente.

Il s'agit d'une zone d'entrée et de sortie de la fonction de rappel. La valeur initiale de cette zone est un pointeur NULL ou des octets NULL.

### **CBDCALLBF (entier signé à 10 chiffres)**

La fonction de rappel est appelée en tant qu'appel de fonction.

Utilisez cette zone pour spécifier un pointeur vers la fonction de rappel.

Vous devez spécifier *CallbackFunction* ou *CallbackName*. Si vous spécifiez les deux, le code anomalie RC2486 est renvoyé.

Si *CallbackName* ou *CallbackFunction* n'est pas défini, l'appel échoue avec le code anomalie RC2486.

Cette option n'est pas prise en charge dans les environnements suivants:

- CICS activé z/OS
- Langages de programmation et compilateurs qui ne prennent pas en charge les références de pointeur de fonction

Dans de telles situations, l'appel échoue avec le code anomalie RC2486.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est un pointeur NULL ou des octets NULL.

### **CBDCALLBN (entier signé à 10 chiffres)**

La fonction de rappel est appelée en tant que programme lié dynamiquement.

Vous devez spécifier *CallbackFunction* ou *CallbackName*. Si vous spécifiez les deux, le code anomalie RC2486 est renvoyé.

Si *CallbackName* ou *CallbackFunction* n'est pas défini sur true, l'appel échoue avec le code anomalie RC2486.

Le module est chargé lorsque la première routine de rappel à utiliser est enregistrée et déchargé lorsque la dernière routine de rappel à utiliser désenregistre.

Sauf indication contraire dans le texte suivant, le nom est aligné à gauche dans la zone, sans blancs imbriqués ; le nom lui-même est complété par des blancs à la longueur de la zone. Dans les descriptions qui suivent, les crochets ([ ]) indiquent des informations facultatives:

#### **IBMi**

Le nom de rappel peut être l'un des formats suivants:

- Programme de bibliothèque "/"
- Bibliothèque "/" ServiceProgram ("FunctionName")

Par exemple, MyLibrary/MyProgram(MyFunction).

Le nom de la bibliothèque peut être \*LIBL. Les noms de bibliothèque et de programme sont limités à un maximum de 10 caractères.

#### **AIX and Linux**

Le nom de rappel est le nom d'un module ou d'une bibliothèque chargeable dynamiquement, suffixé avec le nom d'une fonction résidant dans cette bibliothèque. Le nom de la fonction doit être placé entre parenthèses. Le nom de la bibliothèque peut éventuellement être précédé d'un chemin de répertoire:

```
[path]library(function)
```

Si le chemin n'est pas spécifié, le chemin de recherche système est utilisé.

Le nom est limité à un maximum de 128 caractères.

## Windows

Le nom de rappel est le nom d'une bibliothèque de liens dynamiques, suffixé avec le nom d'une fonction résidant dans cette bibliothèque. Le nom de la fonction doit être placé entre parenthèses. Le nom de la bibliothèque peut éventuellement être précédé d'un chemin de répertoire et d'une unité:

```
[d:][path]library(function)
```

Si l'unité et le chemin ne sont pas spécifiés, le chemin de recherche système est utilisé.

Le nom est limité à un maximum de 128 caractères.

## z/OS

Le nom de rappel est le nom d'un module de chargement qui est valide pour la spécification sur le paramètre EP de la macro LINK ou LOAD.

Le nom est limité à un maximum de 8 caractères.

## z/OS CICS

Le nom de rappel est le nom d'un module de chargement valide pour la spécification sur le paramètre PROGRAM de la macro de la commande EXEC CICS LINK.

Le nom est limité à un maximum de 8 caractères.

Le programme peut être défini comme programme distant à l'aide de l'option REMOTESYTEM de la définition PROGRAM installée ou par le programme de routage dynamique.

La région CICS distante doit être connectée à IBM MQ si le programme doit utiliser des appels API IBM MQ . Notez toutefois que la zone CBCHOBJ de la structure MQCBC n'est pas valide sur un système distant.

Si un incident se produit lors de la tentative de chargement de *CallbackName*, l'un des codes d'erreur suivants est renvoyé à l'application:

- RC2495
- RC2496
- RC2497

Un message est également consigné dans le journal des erreurs contenant le nom du module pour lequel le chargement a été tenté et le code anomalie du système d'exploitation.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est une chaîne nulle ou des blancs.

## **CBDSCALLBT (entier signé à 10 chiffres)**

Il s'agit du type de la fonction de rappel. Les valeurs suivantes sont possibles :

### **CBTMC**

Définit ce rappel en tant que fonction de consommateur de message.

Une fonction de rappel de consommateur de message est appelée lorsqu'un message répondant aux critères de sélection spécifiés est disponible sur un descripteur d'objet et que la connexion est démarrée.

### **CTEH**

Définit ce rappel comme la routine d'événement asynchrone ; il n'est pas piloté pour consommer des messages pour un descripteur.

*Hobj* n'est pas requis sur l'appel MQCB définissant le gestionnaire d'événements et est ignoré s'il est spécifié.

Le gestionnaire d'événements est appelé pour les conditions qui affectent l'ensemble de l'environnement du consommateur de messages. La fonction de consommateur est appelée sans message lorsqu'un événement, par exemple l'arrêt ou la mise au repos d'un gestionnaire de files d'attente ou d'une connexion, se produit. Elle n'est pas appelée pour les conditions spécifiques à un consommateur de message unique, par exemple, RC2016.

Les événements sont distribués à l'application, que la connexion soit démarrée ou arrêtée, sauf dans les environnements suivants:

- Environnement CICS on z/OS
- applications sans unités d'exécution

Si l'appelant ne transmet pas l'une de ces valeurs, l'appel échoue avec le code anomalie RC2483

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est CBTMC.

### **CBDMML (entier signé à 10 chiffres)**

Il s'agit de la longueur en octets du message le plus long qui peut être lu à partir du descripteur et transmis à la routine de rappel. Si un message a une longueur plus longue, la routine de rappel reçoit *MaxMsgLength* octets du message et le code anomalie suivant:

- RC2080 ou
- RC2079 si vous avez spécifié GMATM.

La longueur réelle des messages est indiquée dans la zone «CBCLLEN (entier signé à 10 chiffres)», à la page 1071 de la structure MQCBC.

La valeur spéciale suivante est définie:

### **CBDFM**

La longueur de la mémoire tampon est ajustée par le système pour renvoyer les messages sans troncature.

Si la mémoire disponible est insuffisante pour allouer une mémoire tampon pour recevoir le message, le système appelle la fonction de rappel avec un code anomalie RC2071 .

Si, par exemple, vous demandez une conversion de données et que la mémoire disponible est insuffisante pour convertir les données de message, le message non converti est transmis à la fonction de rappel.

Il s'agit d'une zone d'entrée. La valeur initiale de la zone *MaxMsgLength* est CBDFM.

### **CBDOPT (entier signé à 10 chiffres)**

Structure de descripteur de rappel-Zone Options.

Tout ou partie des éléments suivants peuvent être spécifiés. Pour spécifier plusieurs options, ajoutez les valeurs ensemble (n'ajoutez pas la même constante plusieurs fois) ou combinez les valeurs à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations bit). Les combinaisons qui ne sont pas valides sont notées ; toutes les autres combinaisons sont valides.

### **CBDFQ**

L'appel MQCB échoue si le gestionnaire de files d'attente est à l'état de mise au repos.

Sous z/OS, cette option force également l'échec de l'appel MQCB si la connexion (pour une application CICS ou IMS ) est à l'état de mise au repos.

Spécifiez GMFIQ, dans les options MQGMO transmises à l'appel MQCB, pour envoyer une notification aux consommateurs de message lorsqu'ils sont mis au repos.

**Options de contrôle:** Les options suivantes contrôlent si la fonction de rappel est appelée, sans message, lorsque l'état du consommateur change:

### **CRDC**

La fonction de rappel est appelée avec le type d'appel CBCTRC

.

### **CBDSC**

La fonction de rappel est appelée avec le type d'appel CBCTSC.

### **CBDTC**

La fonction de rappel est appelée avec le type d'appel CBCTTC.

## **CBDDC**

La fonction de rappel est appelée avec le type d'appel CBCTDC.

Pour plus de détails sur ces types d'appel, voir «[CBCCALLT \(entier signé à 10 chiffres\)](#)», à la page [1069](#).

**Option par défaut:** si vous n'avez besoin d'aucune des options décrites, utilisez l'option suivante:

## **CBDNO**

Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. Toutes les options sont définies sur leur valeur par défaut.

CBDNO est défini pour aider la documentation du programme ; il n'est pas prévu que cette option soit utilisée avec d'autres, mais comme sa valeur est nulle, une telle utilisation ne peut pas être détectée.

Il s'agit d'une zone d'entrée. La valeur initiale de la zone *Options* est CBDNO.

## **CBDSID (entier signé à 10 chiffres)**

Structure de descripteur de rappel-Zone StrucId .

Il s'agit de l'identificateur de structure ; la valeur doit être:

## **CBDSI**

Identificateur de la structure de descripteur de rappel.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est CBDSI.

## **CBDVER (entier signé à 10 chiffres)**

Structure du descripteur de rappel-Zone Version.

Il s'agit du numéro de version de la structure ; la valeur doit être:

## **CBDV1**

Structure de descripteur de rappel Version-1 .

La constante suivante indique le numéro de version de la version en cours:

## **CCBDCV**

Version actuelle de la structure de descripteur de rappel.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est CBDV1.

## **Valeurs initiales**

<i>Tableau 688. Zones dans MQCBD</i>		
<b>Nom de zone</b>	<b>Nom de la constante</b>	<b>Valeur de la constante</b>
<i>StrucId</i>	CBDSI	'CBD↵'
<i>Version</i>	CBDV1	1
<i>CallbackType</i>	CBTMC	1
<i>Options</i>	CBDNO	0
<i>CallbackArea</i>	Aucun	Octets nuls
<i>CallbackFunction</i>	Aucun	Octets nuls
<i>CallbackName</i>	Aucun	Espaces vides
<i>MaxMsgLength</i>	CBDFM	-1

### **Remarque :**

1. Le symbole ↵ représente un caractère blanc unique.

## Déclaration RPG

```
D* MQCBD Structure
D*
D*
D* Structure identifier
D  CBDSID          1      4      INZ('CBD ')
D*
D* Structure version number
D  CBDVER          5      8I 0  INZ(1)
D*
D* Callback function type
D  CBDCALLBT       9      12I 0 INZ(1)
D*
** Options controlling message
D* consumption
D  CBDOPT          13     16I 0 INZ(0)
D*
D* User data passed to the function
D  CBDCALLBA       17     32*
D*
D* FP: Callback function pointer
D  CBDCALLBF       33     48*
D*
D* Callback name
D  CBDCALLBN       49     176   INZ('\0')
D*
D* Maximum message length
D  CBDMML          177    180I 0 INZ(-1)
```



## MQCHARV (Variable Length String) sur IBM i

Utilisez la structure MQCHARV pour décrire une chaîne de longueur variable.

### Présentation

**Jeu de caractères et codage:** les données de MQCHARV doivent être dans le codage du gestionnaire de files d'attente local fourni par ENNAT et le jeu de caractères de la zone VCCDP dans la structure. Si l'application s'exécute en tant que IBM MQ MQI client, la structure doit être dans le codage du client. Certains jeux de caractères ont une représentation qui dépend du codage. Si VCCDP est l'un de ces jeux de caractères, le codage utilisé est le même que celui des autres zones du MQCHARV. Le jeu de caractères identifié par VSCCSID peut être un jeu de caractères codé sur deux octets (DBCS).

**Utilisation:** La structure MQCHARV traite les données qui peuvent être discontinuées à la structure qui les contient. Pour traiter ces données, les zones déclarées avec le type de données de pointeur peuvent être utilisées.

- [«Zones», à la page 1080](#)
- [«Valeurs initiales», à la page 1082](#)
- [«Déclaration RPG», à la page 1082](#)
- [«Redéfinition de CSAPL», à la page 1082](#)

### Zones

La structure MQCHARV contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

#### VCCDP (entier signé à 10 chiffres)

Il s'agit de l'identificateur de jeu de caractères de la chaîne de longueur variable adressée par la zone VCHRP ou VCHRO.

La valeur initiale de cette zone est CSAPL. Cette valeur est définie par IBM MQ pour indiquer que le gestionnaire de files d'attente doit la remplacer par l'identificateur de jeu de caractères réel du gestionnaire de files d'attente. Cela est identique au comportement de CSQM. Par conséquent, la valeur CSAPL n'est jamais associée à une chaîne de longueur variable. La valeur initiale de cette zone



peut être modifiée en définissant une valeur différente pour la constante CSAPL de votre unité de compilation par les moyens appropriés pour le langage de programmation de votre application.

### **VCHRL (entier signé à 10 chiffres)**

Longueur en octets de la chaîne de longueur variable adressée par la zone VCHRP ou VCHRO.

La valeur initiale de cette zone est 0. La valeur doit être supérieure ou égale à zéro ou la valeur spéciale suivante qui est reconnue:

#### **VSNTL**

Si VSNTL n'est pas spécifié, les octets VCHRL sont inclus dans la chaîne. Si des caractères nuls sont présents, ils ne délimitent pas la chaîne.

Si VSNTL est spécifié, la chaîne est délimitée par la première valeur null rencontrée dans la chaîne. La valeur null elle-même n'est pas incluse dans cette chaîne.

**Remarque :** Le caractère null utilisé pour terminer une chaîne si VSNTL est spécifié est une valeur null du jeu de codes spécifié par VCCDP.

Par exemple, dans UTF-16 (CCSID 1200, 13488 et 17584), il s'agit du codage Unicode à 2 octets où une valeur null est représentée par un nombre de 16 bits de zéros. En UTF-16, il est courant de rechercher des octets uniques définis sur zéro qui font partie de caractères (caractères ASCII 7 bits par exemple), mais les chaînes ne se terminent par une valeur nulle que lorsque deux octets 'zéro' sont trouvés sur une limite d'octet pair. Il est possible d'obtenir deux octets 'zéro' sur une limite impaire lorsqu'ils font partie de caractères valides. Par exemple, x'01'x'00'x'00'x'00'x'30' représente deux caractères Unicode valides et ne termine pas la chaîne par la valeur null.

### **VCHRO (entier signé à 10 chiffres)**

Décalage en octets de la chaîne de longueur variable à partir du début de MQCHARV ou de la structure qui la contient.

Lorsque la structure MQCHARV est imbriquée dans une autre structure, cette valeur correspond au décalage en octets de la chaîne de longueur variable à partir du début de la structure qui contient cette structure MQCHARV. Lorsque la structure MQCHARV n'est pas imbriquée dans une autre structure, par exemple, si elle est spécifiée en tant que paramètre dans un appel de fonction, le décalage est relatif au début de la structure MQCHARV.

Le décalage peut être positif ou négatif. Vous pouvez utiliser la zone VCHRP ou VCHRO pour indiquer la chaîne de longueur variable, mais pas les deux.

La valeur initiale de cette zone est 0.

### **VCHRP (pointeur)**

Il s'agit d'un pointeur vers la chaîne de longueur variable.

Vous pouvez utiliser la zone VCHRP ou VCHRO pour indiquer la chaîne de longueur variable, mais pas les deux.

La valeur initiale de cette zone est un pointeur NULL ou des octets NULL.

### **VCHRS (entier signé à 10 chiffres)**

Taille en octets de la mémoire tampon adressée par la zone VCHRP ou VCHRO.

Lorsque la structure MQCHARV est utilisée comme zone de sortie dans un appel de fonction, cette zone doit être initialisée avec la longueur de la mémoire tampon fournie. Si la valeur de VCHRL est supérieure à VCHRS, seuls les octets de données VCHRS seront renvoyés à l'appelant dans la mémoire tampon.

La valeur doit être supérieure ou égale à zéro ou la valeur spéciale suivante reconnue:

#### **VSUSL**

Si VSUSL est spécifié, la longueur de la mémoire tampon est extraite de la zone VCHRL dans la structure MQCHARV. Cette valeur spéciale n'est pas appropriée lorsque la structure est utilisée

comme zone de sortie et qu'une mémoire tampon est fournie. Il s'agit de la valeur initiale de cette zone.

## Valeurs initiales

Tableau 689. Valeurs initiales MQCHARV pour les constantes		
Nom de zone	Nom de la constante	Valeur de la constante
VCHRP	Aucun	Pointeur NULL ou octets NULL.
VCHRO	Aucun	0
VCHRS	VSUSL	-1
VCHRL	Aucun	0
VCHRC	CSAPL	-3

## Déclaration RPG

```

D* .1.....2.....3.....4.....5.....6.....7..
D* MQCHARV Structure
D*
D* Address of variable length string
D VCHRP          1      16*
D* Offset of variable length string
D VCHRO         17      20I 0
D* Size of buffer
D VCHRS         21      24I 0
D* Length of variable length string
D VCHRL         25      28I 0
D* CCSID of variable length string
D VCHRC         29      32I 0

```

## Redéfinition de CSAPL

Contrairement aux langages de programmation pris en charge sur d'autres plateformes, RPG ne dispose pas d'un moyen de redéfinir une constante définie. Vous devez donc définir chaque VCCDP spécifiquement si vous souhaitez utiliser une valeur autre que CSAPL.

## IBM i MQCIH (en-tête CICS bridge) sous IBM i

La structure MQCIH décrit les informations qui peuvent être présentes au début d'un message envoyé à CICS bridge via IBM MQ for z/OS.

## Présentation

**Nom de format:** FMCICS.

**Versión:** la version actuelle de MQCIH est CIVER2. Les zones qui existent uniquement dans la version la plus récente de la structure sont identifiées comme telles dans les descriptions qui suivent.

Le fichier COPY fourni contient la version la plus récente de MQCIH, avec la valeur initiale de la zone CIVER définie sur CIVER2.

**Jeu de caractères et codage:** des conditions spéciales s'appliquent au jeu de caractères et au codage utilisés pour la structure MQCIH et les données de message d'application:

- Les applications qui se connectent au gestionnaire de files d'attente qui possède la file d'attente CICS bridge doivent fournir une structure MQCIH qui se trouve dans le jeu de caractères et le codage du gestionnaire de files d'attente. En effet, la conversion de données de la structure MQCIH n'est pas effectuée dans ce cas.

- Les applications qui se connectent à d'autres gestionnaires de files d'attente peuvent fournir une structure MQCIH qui se trouve dans l'un des jeux de caractères et codages pris en charge ; la conversion de MQCIH est effectuée par l'agent de canal de réception des messages connecté au gestionnaire de files d'attente qui possède la file d'attente CICS bridge .

**Remarque :** Il y a une exception à cela. Si le gestionnaire de files d'attente propriétaire de la file d'attente CICS bridge utilise CICS pour la mise en file d'attente répartie, le MQCIH doit se trouver dans le jeu de caractères et le codage du gestionnaire de files d'attente propriétaire de la file d'attente CICS bridge .

- Les données de message d'application qui suivent la structure MQCIH doivent être dans le même jeu de caractères et dans le même codage que la structure MQCIH. Les zones *CICSI* et *CIENC* de la structure MQCIH ne peuvent pas être utilisées pour spécifier le jeu de caractères et le codage des données de message d'application.

Un exit de conversion de données doit être fourni par l'utilisateur pour convertir les données de message d'application si les données ne sont pas l'un des formats intégrés pris en charge par le gestionnaire de files d'attente.

**Syntaxe:** si les valeurs requises par l'application sont identiques aux valeurs initiales affichées dans [Tableau 691](#), à la page 1092 et que le pont s'exécute avec AUTH=LOCAL ou AUTH=IDENTIFY, la structure MQCIH peut être omise du message. Dans tous les autres cas, la structure doit être présente.

Le pont accepte une structure MQCIH version-1 ou version-2 , mais pour les transactions 3270, une structure version-2 doit être utilisée.

L'application doit s'assurer que les zones documentées en tant que zones de "demande" ont des valeurs appropriées dans le message envoyé au pont ; ces zones sont des entrées pour le pont.

Les zones documentées en tant que zones de "réponse" sont définies par CICS bridge dans le message de réponse que le pont envoie à l'application. Les informations d'erreur sont renvoyées dans les zones *CIRET*, *CIFNC*, *CICC*, *CIREA* et *CIAC* , mais elles ne sont pas toutes définies dans tous les cas. [Tableau 690](#), à la page 1083 indique les zones qui sont définies pour différentes valeurs de *CIRET*.

<i>Tableau 690. Contenu des zones d'informations d'erreur dans la structure MQCIH</i>				
<b>CIRET</b>	<b>CIFNC</b>	<b>CICC</b>	<b>CIREA</b>	<b>CIAC</b>
CRC000	-	-	-	-
CRC003	-	-	FBC*	-
CRC002 CRC008	Nom d'appel IBM MQ	IBM MQ <i>CMPCOD</i>	IBM MQ <i>REASON</i>	-
CRC001 CRC006 CRC007 CRC009	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	-
CRC004 CRC005	-	-	-	CICS ABCODE

- «Zones», à la page 1083
- «Valeurs initiales», à la page 1092
- «Déclaration RPG», à la page 1093

## Zones

La structure MQCIH contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

### **CIAC (chaîne de caractères de 4 octets)**

Code de fin anormale.

La valeur renvoyée dans cette zone est significative uniquement si la zone *CIRET* a la valeur CRC005 ou CRC004. Si tel est le cas, *CIAC* contient la valeur ABCODE CICS .

Il s'agit d'une zone de réponse. La longueur de cette zone est indiquée par LNABNC. La valeur initiale de cette zone est de 4 caractères blancs.

Il s'agit d'un indicateur spécifiant si les descripteurs ADS doivent être envoyés sur les demandes SEND et RECEIVE BMS. Les valeurs suivantes sont définies :

**ADAUCUN**

N'envoyez ou ne recevez pas de descripteur ADS.

**ADSEND**

Descripteur ADS d'envoi.

**ADRECV**

Descripteur ADS de réception.

**ADMSGF**

Utilisez le format de message pour le descripteur ADS.

Cela provoque l'envoi ou la réception du descripteur ADS à l'aide de la forme longue du descripteur ADS. Le format long comporte des zones alignées sur des limites de 4 octets.

La zone *CIADS* doit être définie comme suit:

- Si les descripteurs ADS ne sont pas utilisés, définissez la zone sur *ADNONE*.
- Si des descripteurs ADS *sont* utilisés et avec le *même* CCSID dans chaque environnement, définissez la zone sur la somme d'ADSEND et d'ADRECV.
- Si des descripteurs ADS *sont* utilisés, mais avec des CCSID *différents* dans chaque environnement, définissez la zone sur la somme de ADSEND, ADRECV et ADMSGF.

Il s'agit d'une zone de demande utilisée uniquement pour les transactions 3270. La valeur initiale de cette zone est *ADNONE*.

**CIADS (entier signé à 10 chiffres)**

Descripteur ADS d'envoi/réception.

Il s'agit d'un indicateur spécifiant si les descripteurs ADS doivent être envoyés sur les demandes SEND et RECEIVE BMS. Les valeurs suivantes sont définies :

**ADAUCUN**

N'envoyez ou ne recevez pas de descripteur ADS.

**ADSEND**

Descripteur ADS d'envoi.

**ADRECV**

Descripteur ADS de réception.

**ADMSGF**

Utilisez le format de message pour le descripteur ADS.

Cela provoque l'envoi ou la réception du descripteur ADS à l'aide de la forme longue du descripteur ADS. Le format long comporte des zones alignées sur des limites de 4 octets.

La zone *CIADS* doit être définie comme suit:

- Si les descripteurs ADS ne sont pas utilisés, définissez la zone sur *ADNONE*.
- Si des descripteurs ADS *sont* utilisés et avec le *même* CCSID dans chaque environnement, définissez la zone sur la somme d'ADSEND et d'ADRECV.
- Si des descripteurs ADS *sont* utilisés, mais avec des CCSID *différents* dans chaque environnement, définissez la zone sur la somme de ADSEND, ADRECV et ADMSGF.

Il s'agit d'une zone de demande utilisée uniquement pour les transactions 3270. La valeur initiale de cette zone est *ADNONE*.

**CIAI (chaîne de caractères de 4 octets)**

Touche AID.

Il s'agit de la valeur initiale de la clé AID lorsque la transaction est démarrée. Il s'agit d'une valeur à 1 octet, alignée à gauche.

Il s'agit d'une zone de demande utilisée uniquement pour les transactions 3270. La longueur de cette zone est indiquée par LNATID. La valeur initiale de cette zone est 4 blancs.

#### **CIAUT (chaîne de caractères de 8 octets)**

Mot de passe ou mot de passe.

Il s'agit d'un mot de passe ou d'un mot de passe. Si l'authentification par identificateur d'utilisateur est active pour CICS bridge, *CIAUT* est utilisé avec l'identificateur d'utilisateur dans le contexte d'identité MQMD pour authentifier l'expéditeur du message.

Il s'agit d'une zone de demande. La longueur de cette zone est indiquée par LNAUTH. La valeur initiale de cette zone est 8 blancs.

#### **CICC (entier signé à 10 chiffres)**

Code achèvement IBM MQ ou CICS EIBRESP.

La valeur renvoyée dans cette zone dépend de *CIRET* ; voir [Tableau 690](#), à la page 1083.

Il s'agit d'une zone de réponse. La valeur initiale de ce champ est CCOK.

#### **CICNC (chaîne de caractères de 4 octets)**

Code de transaction de fin anormale.

Il s'agit du code de fin anormale à utiliser pour mettre fin à la transaction (normalement une transaction conversationnelle qui demande plus de données). Sinon, cette zone est mise à blanc.

Il s'agit d'une zone de demande utilisée uniquement pour les transactions 3270. La longueur de cette zone est indiquée par LNCNCL. La valeur initiale de cette zone est 4 blancs.

#### **CICP (entier signé à 10 chiffres)**

Position du curseur.

Il s'agit de la position initiale du curseur lorsque la transaction est démarrée. Par la suite, pour les transactions conversationnelles, la position du curseur est dans le vecteur RECEIVE.

Il s'agit d'une zone de demande utilisée uniquement pour les transactions 3270. La valeur initiale de cette zone est 0. Cette zone n'est pas présente si *CIVER* est inférieur à *CIVER2*.

#### **CICSI (entier signé à 10 chiffres)**

Réservé.

Il s'agit d'une zone réservée ; sa valeur n'est pas significative. La valeur initiale de cette zone est 0.

#### **CICT (entier signé à 10 chiffres)**

Indique si la tâche peut être conversationnelle.

Il s'agit d'un indicateur spécifiant si la tâche doit être autorisée à émettre des demandes d'informations supplémentaires ou à se terminer de manière anormale. Il doit s'agir de l'une des valeurs suivantes :

##### **CTYES**

La tâche est conversationnelle.

##### **NCTN**

La tâche n'est pas conversationnelle.

Il s'agit d'une zone de demande utilisée uniquement pour les transactions 3270. La valeur initiale de cette zone est CTNO.

#### **CIENC (entier signé à 10 chiffres)**

Réservé.

Il s'agit d'une zone réservée ; sa valeur n'est pas significative. La valeur initiale de cette zone est 0.

### **CIEO (entier signé à 10 chiffres)**

Décalage de l'erreur dans le message.

Il s'agit de la position des données non valides détectées par l'exit de pont. Cette zone indique le décalage entre le début du message et l'emplacement des données non valides.

Il s'agit d'une zone de réponse utilisée uniquement pour les transactions 3270. La valeur initiale de cette zone est 0. Cette zone n'est pas présente si *CIVER* est inférieur à *CIVER2*.

### **CIFAC (chaîne de bits de 8 octets)**

Jeton de fonction de pont.

Il s'agit d'un jeton de fonction de pont de 8 octets. Le but d'un jeton de fonction de pont est d'autoriser plusieurs transactions dans une pseudo-conversation à utiliser la même fonction de pont (terminal 3270 virtuel). Dans le premier ou le seul message d'une pseudo-conversation, une valeur de FCNONE doit être définie ; cela indique à CICS d'allouer une nouvelle fonction de pont pour ce message. Un jeton de fonction de pont est renvoyé dans les messages de réponse lorsqu'un *CIFKT* différent de zéro est spécifié dans le message d'entrée. Les messages d'entrée suivants peuvent alors utiliser le même jeton de fonction de pont.

La valeur spéciale suivante est définie:

#### **FCNONE**

Aucun jeton BVT spécifié.

Il s'agit d'une zone de demande et de réponse utilisée uniquement pour les transactions 3270. La longueur de cette zone est donnée par *LNFAAC*. La valeur initiale de cette zone est FCNONE.

### **CIFKT (entier signé à 10 chiffres)**

Heure de libération de la fonction de pont.

Il s'agit de la durée, en secondes, pendant laquelle la fonction de pont sera conservée après la fin de la transaction utilisateur. Pour les transactions non conversationnelles, la valeur doit être zéro.

Il s'agit d'une zone de demande utilisée uniquement pour les transactions 3270. La valeur initiale de cette zone est 0.

### **CIFL (chaîne de caractères de 4 octets)**

Attributs de terminal émulé.

Il s'agit du nom d'un terminal installé qui doit être utilisé comme modèle pour la fonction de pont. Une valeur vide signifie que *CIFL* est extraite de la définition de profil de transaction de pont ou qu'une valeur par défaut est utilisée.

Il s'agit d'une zone de demande utilisée uniquement pour les transactions 3270. La longueur de cette zone est donnée par *LNFACL*. La valeur initiale de cette zone est 4 blancs.

### **CIFLG (entier signé à 10 chiffres)**

Indicateurs.

La valeur doit être:

#### **CIFNON**

Aucun indicateur.

Il s'agit d'une zone de demande. La valeur initiale de cette zone est CIFNON.

### **CIFMT (chaîne de caractères de 8 octets)**

Nom de format IBM MQ des données qui suivent MQCIH.

Indique le nom de format IBM MQ des données qui suivent la structure MQCIH.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de la zone *MDFMT* dans MQMD.

Ce nom de format est également utilisé pour le message de réponse, si la zone *CIRFM* a la valeur FMNONE.

- Pour les demandes DPL, *CIFMT* doit être le nom de format de la zone de communication.
- Pour les demandes 3270, *CIFMT* doit être CSQCBDCI et *CIRFM* doit être CSQCBDCO.

Les exits de conversion de données pour ces formats doivent être installés sur le gestionnaire de files d'attente où ils doivent être exécutés.

Si le message de demande génère un message de réponse d'erreur, le nom de format du message de réponse d'erreur est FMSTR.

Il s'agit d'une zone de demande. La longueur de cette zone est indiquée par LNFMT. La valeur initiale de cette zone est FMNONE.

### **CIFNC (chaîne de caractères de 4 octets)**

Nom d'appel IBM MQ ou fonction EIBFN CICS.

La valeur renvoyée dans cette zone dépend de *CIRET*; voir [Tableau 690, à la page 1083](#). Les valeurs suivantes sont possibles lorsque *CIFNC* contient un nom d'appel IBM MQ :

#### **CFCONN**

Appel MQCONN.

#### **Commande CFGET**

Appel MQGET.

#### **CFINQ**

Appel MQINQ.

#### **CFOPEN**

Appel MQOPEN.

#### **Instruction CFPUT**

Appel MQPUT.

#### **CFPUT1**

Appel MQPUT1.

#### **CFAUCUN**

Pas d'appel.

Il s'agit d'une zone de réponse. La longueur de cette zone est donnée par LNFUNC. La valeur initiale de cette zone est CFNONE.

### **CIGWI (entier signé à 10 chiffres)**

Intervalle d'attente pour l'appel MQGET émis par la tâche de pont.

Cette zone n'est applicable que si *CIUOW* a la valeur CUFRST. Elle permet à l'application émettrice de spécifier la durée approximative, en millisecondes, pendant laquelle les appels MQGET émis par le pont doivent attendre la seconde demande et les messages de demande suivants pour l'unité de travail démarrée par ce message. Cette option remplace l'intervalle d'attente par défaut utilisé par le pont. Les valeurs spéciales suivantes peuvent être utilisées:

#### **WIDFLT**

Intervalle d'attente par défaut.

Cela entraîne l'attente de CICS bridge pendant la période spécifiée lors du démarrage du pont.

#### **WIULIM**

Intervalle d'attente illimité.

Il s'agit d'une zone de demande. La valeur initiale de cette zone est WIDFLT.

### **CIII (entier signé à 10 chiffres)**

Réservé.

Il s'agit d'une zone réservée. La valeur doit être 0. Cette zone n'est pas présente si *CIVER* est inférieur à *CIVER2*.

### **CILEN (entier signé à 10 chiffres)**

Longueur de la structure MQCIH.

Il doit s'agir de l'une des valeurs suivantes :

#### **CILEN1**

Longueur de la structure d'en-tête d'information version-1 CICS .

#### **CILEN2**

Longueur de la structure d'en-tête d'information version-2 CICS .

La constante suivante indique la longueur de la version en cours:

#### **CILENC**

Longueur de la version actuelle de la structure d'en-tête d'informations CICS .

Il s'agit d'une zone de demande. La valeur initiale de cette zone est *CILEN2*.

### **CILT (entier signé à 10 chiffres)**

Type de lien.

Indique le type d'objet que le pont doit tenter de lier. Il doit s'agir de l'une des valeurs suivantes :

#### **LTPROG**

Programme DPL.

#### **LTTRAN**

Transaction 3270.

Il s'agit d'une zone de demande. La valeur initiale de cette zone est *LTPROG*.

### **CINTI (chaîne de caractères de 4 octets)**

Transaction suivante à joindre.

Il s'agit du nom de la transaction suivante renvoyée par la transaction utilisateur (généralement par *EXEC CICS RETURN TRANSID*). S'il n'y a pas de transaction suivante, cette zone est mise à blanc.

Il s'agit d'une zone de réponse utilisée uniquement pour les transactions 3270. La longueur de cette zone est indiquée par *LNTRID*. La valeur initiale de cette zone est 4 blancs.

### **CIODL (entier signé à 10 chiffres)**

Longueur des données *COMMAREA* de sortie.

Il s'agit de la longueur des données utilisateur à renvoyer au client dans un message de réponse. Cette longueur inclut le nom de programme à 8 octets. La longueur de la zone de communication transmise au programme lié est la longueur maximale de cette zone et la longueur des données utilisateur du message de demande, moins 8.

**Remarque :** La longueur des données utilisateur dans un message correspond à la longueur du message *excluant* la structure *MQCIH*.

Si la longueur des données utilisateur dans le message de demande est inférieure à *CIODL*, l'option *DATALENGTH* de la commande *LINK* est utilisée ; cela permet à *LINK* d'être livré efficacement à une autre région CICS .

La valeur spéciale suivante peut être utilisée:

#### **OLINPT**

La longueur de sortie est identique à la longueur d'entrée.



Cette valeur peut être nécessaire même si aucune réponse n'est demandée, afin de s'assurer que la zone de communication transmise au programme lié est de taille suffisante.

Il s'agit d'une zone de demande utilisée uniquement pour les programmes DPL. Valeur initiale de cette zone OLINPT.

### **CIREA (entier signé à 10 chiffres)**

IBM MQ raison ou code retour, ou CICS EIBRESP2.

La valeur renvoyée dans cette zone dépend de *CIRET* ; voir [Tableau 690](#), à la page 1083.

Il s'agit d'une zone de réponse. La valeur initiale de cette zone est RCNONE.

### **CIRET (entier signé à 10 chiffres)**

Code retour du pont.

Il s'agit du code retour du CICS bridge décrivant le résultat du traitement effectué par le pont. Les zones *CIFNC*, *CICC*, *CIREA* et *CIAC* peuvent contenir des informations supplémentaires (voir [Tableau 690](#), à la page 1083). La valeur est l'une des suivantes :

#### **CRC000**

(0, X'000') Aucune erreur.

#### **CRC001**

(1, X'001') L'instruction EXEC CICS a détecté une erreur.

#### **CRC002**

(2, X'002') L'appel à IBM MQ a détecté une erreur.

#### **CRC003**

(3, X'003') CICS bridge a détecté une erreur.

#### **CRC004**

(4, X'004') CICS bridge s'est arrêté de manière anormale.

#### **CRC005**

(5, X'005') L'application s'est terminée de manière anormale.

#### **CRC006**

(6, X'006') Une erreur de sécurité s'est produite.

#### **CRC007**

(7, X'007') Programme non disponible.

#### **CRC008**

(8, X'008') Deuxième message ou message ultérieur dans l'unité d'oeuvre en cours qui n'a pas été reçu dans le délai imparti.

#### **CRC009**

(9, X'009') Transaction non disponible.

Il s'agit d'une zone de réponse. La valeur initiale de cette zone est CRC000.

### **CIRFM (chaîne de caractères de 8 octets)**

Nom de format IBM MQ du message de réponse.

Il s'agit du nom de format IBM MQ du message de réponse qui sera envoyé en réponse au message en cours. Les règles de codage sont les mêmes que celles de la zone *MDFMT* dans MQMD.

Il s'agit d'une zone de demande utilisée uniquement pour les programmes DPL. La longueur de cette zone est indiquée par LNFMT. La valeur initiale de cette zone est FMNONE.

### **CIRSI (chaîne de caractères de 4 octets)**

Réservé.

Il s'agit d'une zone réservée. La valeur doit être à 4 blancs. La longueur de cette zone est donnée par LNRSID.

**CIRS1 (chaîne de caractères de 8 octets)**

Réservé.

Il s'agit d'une zone réservée. La valeur doit être à 8 blancs.

**CIRS2 (chaîne de caractères de 8 octets)**

Réservé.

Il s'agit d'une zone réservée. La valeur doit être à 8 blancs.

**CIRS3 (chaîne de caractères de 8 octets)**

Réservé.

Il s'agit d'une zone réservée. La valeur doit être à 8 blancs.

**CIRS4 (entier signé à 10 chiffres)**

Réservé.

Il s'agit d'une zone réservée. La valeur doit être 0. Cette zone n'est pas présente si *CIVER* est inférieur à *CIVER2*.

**CIRTI (chaîne de caractères de 4 octets)**

Réservé.

Il s'agit d'une zone réservée. La valeur doit être à 4 blancs. La longueur de cette zone est indiquée par *LNTRID*.

**CISC (chaîne de caractères de 4 octets)**

Code de début de transaction.

Il s'agit d'un indicateur spécifiant si le pont émule une transaction de terminal ou une transaction *START*. Il doit s'agir de l'une des valeurs suivantes :

**SCSTRT**

Début.

**SCDATA**

Données de démarrage.

**SCTERM**

Mettez fin à l'entrée.

**SCNONE**

Néant.

Dans la réponse du pont, cette zone est définie sur le code de début correspondant à l'ID transaction suivant contenu dans la zone *CINTI* . Les codes de début suivants sont possibles dans la réponse:

- SCSTRT
- SCDATA
- SCTERM

Pour CICS Transaction Server 1.2 , cette zone est une zone de demande uniquement ; sa valeur dans la réponse n'est pas définie.

Pour CICS Transaction Server 1.3 et les éditions suivantes, il s'agit à la fois d'une zone de demande et d'une zone de réponse.

Cette zone est utilisée uniquement pour les transactions 3270. La longueur de cette zone est indiquée par *LNSTCO*. La valeur initiale de cette zone est *SCNONE*.

**CISID (chaîne de caractères de 4 octets)**

Identificateur de structure.

La valeur doit être:

**CISIDV**

Identificateur de la structure d'en-tête d'informations CICS .

Il s'agit d'une zone de demande. La valeur initiale de cette zone est CISIDV.

**CITES (entier signé à 10 chiffres)**

Statut à la fin de la tâche.

Cette zone affiche le statut de la transaction utilisateur à la fin de la tâche. L'une des valeurs suivantes est renvoyée:

**TENOSY**

Non synchronisé.

La transaction utilisateur n'est pas encore terminée et n'est pas synchronisée. La zone *MDMT* de MQMD est MTRQST dans ce cas.

**TECMIT**

Validation de l'unité de travail.

La transaction utilisateur n'est pas encore terminée, mais elle a synchronisé la première unité de travail. La zone *MDMT* de MQMD est MTDGRM dans ce cas.

**TEBACK**

Unité de travail d'arrière-système.

La transaction utilisateur n'est pas encore terminée. L'unité de travail en cours sera annulée. La zone *MDMT* de MQMD est MTDGRM dans ce cas.

**TEENDT**

Fin de la tâche.

La transaction utilisateur s'est arrêtée (ou s'est arrêtée de manière anormale). La zone *MDMT* dans MQMD est MTRPLY dans ce cas.

Il s'agit d'une zone de réponse utilisée uniquement pour les transactions 3270. La valeur initiale de cette zone est TENOSY.

**CITI (chaîne de caractères de 4 octets)**

Transaction à joindre.

Si *CILT* a la valeur LTRAN, *CITI* est l'identificateur de transaction de la transaction utilisateur à exécuter ; une valeur non vide doit être spécifiée dans ce cas.

Si *CILT* a la valeur LTPROG, *CITI* est le code de transaction sous lequel tous les programmes de l'unité de travail doivent être exécutés. Si la valeur spécifiée est vide, le code de transaction par défaut du pont DPL CICS (CKBP) est utilisé. Si la valeur n'est pas vide, elle doit avoir été définie sur CICS en tant que TRANSACTION locale avec un programme initial CSQCBP00. Cette zone est applicable uniquement lorsque *CIUOW* a la valeur CUFRST ou CUONLY.

Il s'agit d'une zone de demande. La longueur de cette zone est indiquée par LNTRID. La valeur initiale de cette zone est 4 blancs.

**CIUOW (entier signé à 10 chiffres)**

Contrôle de l'unité de travail.

Contrôle le traitement de l'unité de travail effectué par CICS bridge. Vous pouvez demander au pont d'exécuter une seule transaction ou un ou plusieurs programmes au sein d'une unité de travail. La zone indique si le CICS bridge doit démarrer une unité de travail, exécuter la fonction demandée dans l'unité de travail en cours ou mettre fin à l'unité de travail en la validant ou en la sauvegardant. Diverses combinaisons sont prises en charge, pour optimiser les flux de transmission de données.

Il doit s'agir de l'une des valeurs suivantes :

**CUONLY**

Démarrez l'unité d'oeuvre, exécutez la fonction, puis validez l'unité d'oeuvre (DPL et 3270).

**CUCONT**

Données supplémentaires pour l'unité d'oeuvre en cours (3270 uniquement).

**CUFRST**

Démarrer l'unité d'oeuvre et exécuter la fonction (DPL uniquement).

**CUMIDL**

Exécuter la fonction dans l'unité d'oeuvre en cours (DPL uniquement).

**CULAST**

Exécutez la fonction, puis validez l'unité de travail (DPL uniquement).

**CUCMIT**

Validez l'unité d'oeuvre (DPL uniquement).

**CUBACK**

Revenez à l'unité d'oeuvre (DPL uniquement).

Il s'agit d'une zone de demande. La valeur initiale de cette zone est CUONLY.

**CIVER (entier signé à 10 chiffres)**

Numéro de version de la structure.

Il doit s'agir de l'une des valeurs suivantes :

**CIVER1**

Structure d'en-tête d'information Version-1 CICS .

**CIVER2**

Structure d'en-tête d'information Version-2 CICS .

Les zones qui existent uniquement dans la version la plus récente de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

**CiC-C**

Version actuelle de la structure d'en-tête d'informations CICS .

Il s'agit d'une zone de demande. La valeur initiale de cette zone est CIVER2.

**Valeurs initiales**

<i>Tableau 691. Zones dans MQCIH</i>		
<b>Nom de zone</b>	<b>Nom de la constante</b>	<b>Valeur de la constante</b>
<i>CISID</i>	CISIDV	' CIH~ '
<i>CIVER</i>	CIVER2	2
<i>CILEN</i>	CILEN2	180
<i>CIENC</i>	Aucun	0
<i>CICSI</i>	Aucun	0
<i>CIFMT</i>	FMAUCUN	Espaces vides
<i>CIFLG</i>	CIFNON	0
<i>CIRET</i>	CRC000	0
<i>CICC</i>	CCOK	0
<i>CIREA</i>	RCNONE	0
<i>CIUOW</i>	CUONLY	273
<i>CIGWI</i>	WIDFLT	-2

Tableau 691. Zones dans MQCIH (suite)

Nom de zone	Nom de la constante	Valeur de la constante
<i>CILT</i>	LTPROG	1
<i>CIODL</i>	OLINPT	-1
<i>CIFKT</i>	Aucun	0
<i>CIADS</i>	ADAUCUN	0
<i>CICT</i>	NCTN	0
<i>CITES</i>	TENOSY	0
<i>CIFAC</i>	FCNONE	Valeurs NULL
<i>CIFNC</i>	CFAUCUN	Espaces vides
<i>CIAC</i>	Aucun	Espaces vides
<i>CIAUT</i>	Aucun	Espaces vides
<i>CIRS1</i>	Aucun	Espaces vides
<i>CIRFM</i>	FMAUCUN	Espaces vides
<i>CIRSI</i>	Aucun	Espaces vides
<i>CIRTI</i>	Aucun	Espaces vides
<i>CITI</i>	Aucun	Espaces vides
<i>CIFL</i>	Aucun	Espaces vides
<i>CIAI</i>	Aucun	Espaces vides
<i>CISC</i>	SCNONE	Espaces vides
<i>CICNC</i>	Aucun	Espaces vides
<i>CINTI</i>	Aucun	Espaces vides
<i>CIRS2</i>	Aucun	Espaces vides
<i>CIRS3</i>	Aucun	Espaces vides
<i>CICP</i>	Aucun	0
<i>CIEO</i>	Aucun	0
<i>CIII</i>	Aucun	0
<i>CIRS4</i>	Aucun	0

**Remarques :**

1. Le symbole - représente un caractère blanc unique.

**Déclaration RPG**

```

D*..1.....2.....3.....4.....5.....6.....7..
D* MQCIH Structure
D*
D* Structure identifier
D  CISID          1      4  INZ('CIH ')
D* Structure version number
D  CIVER          5      8I 0 INZ(2)
D* Length of MQCIH structure
    
```

```

D CILEN                9      12I 0 INZ(180)
D* Reserved
D CIENC                13     16I 0 INZ(0)
D* Reserved
D CICSI               17     20I 0 INZ(0)
D* MQ format name of data that followsMQCIH
D CIFMT               21     28   INZ('      ')
D* Flags
D CIFLG               29     32I 0 INZ(0)
D* Return code from bridge
D CIRET               33     36I 0 INZ(0)
D* MQ completion code or CICSEIBRESP
D CIIC                37     40I 0 INZ(0)
D* MQ reason or feedback code, or CICSEIBRESP2
D CIREA               41     44I 0 INZ(0)
D* Unit-of-work control
D CIUOW               45     48I 0 INZ(273)
D* Wait interval for MQGET call issuedby bridge task
D CIGWI               49     52I 0 INZ(-2)
D* Link type
D CILT                53     56I 0 INZ(1)
D* Output COMMAREA data length
D CIODL               57     60I 0 INZ(-1)
D* Bridge facility release time
D CIFKT               61     64I 0 INZ(0)
D* Send/receive ADS descriptor
D CIADS               65     68I 0 INZ(0)
D* Whether task can beconversational
D CICT                69     72I 0 INZ(0)
D* Status at end of task
D CITES               73     76I 0 INZ(0)
D* Bridge facility token
D CIFAC               77     84   INZ(X'00000000000000-
D                      00')
D* MQ call name or CICS EIBFNfunction
D CIFNC               85     88   INZ('      ')
D* Abend code
D CIAC                89     92   INZ
D* Password or passticket
D CIAUT               93    100   INZ
D* Reserved
D CIRS1              101    108   INZ
D* MQ format name of reply message
D CIRFM              109    116   INZ('      ')
D* Remote CICS system ID to use
D CIRSI              117    120   INZ
D* CICS RTRANSID to use
D CIRTI              121    124   INZ
D* Transaction to attach
D CITI               125    128   INZ
D* Terminal emulated attributes
D CIFL               129    132   INZ
D* AID key
D CIAI               133    136   INZ
D* Transaction start code
D CISC               137    140   INZ('      ')
D* Abend transaction code
D CICNC              141    144   INZ
D* Next transaction to attach
D CINTI              145    148   INZ
D* Reserved
D CIRS2              149    156   INZ
D* Reserved
D CIRS3              157    164   INZ
D* Cursor position
D CICP               165    168I 0 INZ(0)
D* Offset of error in message
D CIEO               169    172I 0 INZ(0)
D* Reserved
D CIII               173    176I 0 INZ(0)
D* Reserved
D CIRS4              177    180I 0 INZ(0)
D*

```

## IBM i MQCMHO (Create message handle options) sous IBM i

La structure **MQCMHO** permet aux applications de spécifier des options qui contrôlent la façon dont les descripteurs de message sont créés.

## Présentation

### Objet

La structure est un paramètre d'entrée dans l'appel **MQCRTMH** .

### Jeu de caractères et codage

Les données dans **MQCMHO** doivent être dans le jeu de caractères de l'application et dans le codage de l'application (ENNAT).

- [«Zones», à la page 1095](#)
- [«Valeurs initiales», à la page 1096](#)
- [«Déclaration RPG», à la page 1097](#)

## Zones

La structure MQCMHO contient les zones suivantes ; les zones sont décrites par ordre alphabétique:

### CMOPT (entier signé à 10 chiffres)

L'une des options suivantes peut être spécifiée:

#### CMVAL

Lorsque **MQSETMP** est appelé pour définir une propriété dans ce descripteur de message, le nom de la propriété est validé pour s'assurer qu'il:

- ne contient pas de caractères non valides.
- ne commence pas par "JMS" ou "usr.JMS", à l'exception de ce qui suit:
  - JMSCorrelationID
  - JMSReplyTo
  - JMSType
  - JMSXGroupID
  - JMSXGroupSeq

Ces noms sont réservés aux propriétés JMS .

- n'est pas l'un des mots clés suivants, dans un mélange de majuscules ou de minuscules:
  - "ET"
  - "ENTRE"
  - "ECHAPPEMENT"
  - "FAUX"
  - "DANS"
  - "IS"
  - « Comme »
  - "NON"
  - "NULL"
  - "OU"
  - "VRAI"
- ne commence pas par "Corps". ou "Racine". (sauf pour "Root.MQMD").

Si la propriété est MQ-defined ("mq.\*") et que le nom est reconnu, les zones de descripteur de propriété sont définies sur les valeurs correctes pour la propriété. Si la propriété n'est pas reconnue, la zone *Support* du descripteur de propriété est définie sur **PDSUPO** (pour plus d'informations, voir [PDSUP](#) ).

## CMDEFV

Indique que le niveau de validation par défaut des noms de propriété est atteint.

Le niveau de validation par défaut est équivalent à celui spécifié par **CMVAL**.

Dans une édition ultérieure, une option d'administration pourra être définie, ce qui modifiera le niveau de validation qui sera appliqué lorsque **CMDEFV** sera défini.

Il s'agit de la valeur par défaut.

## Analyse de la variance

Aucune validation n'est effectuée sur le nom de la propriété. Voir la description de **CMVAL**.

**Option par défaut:** Si aucune des options décrites précédemment dans cette section n'est requise, l'option suivante peut être utilisée:

## CMNONE

Toutes les options prennent leurs valeurs par défaut. Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. **CMNONE** aide la documentation du programme ; il n'est pas prévu que cette option soit utilisée avec d'autres, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **CMDEFV**.

## CMSID (entier signé à 10 chiffres)

Il s'agit de l'identificateur de structure ; la valeur doit être:

### CMSIDV

Identificateur de la structure des options de création de descripteur de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **CMSIDV**.

## CMVER (entier signé à 10 chiffres)

Il s'agit du numéro de version de la structure ; la valeur doit être:

### CMVER1

Version-1 crée une structure d'options de descripteur de message.

La constante suivante indique le numéro de version de la version en cours:

### CMVERC

Version actuelle de la structure des options de création de descripteur de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **CMVER1**.

## Valeurs initiales

Nom de zone	Nom de la constante	Valeur de la constante
<i>CMSID</i>	CMSIDV	' CMHO '
<i>CMVER</i>	CMVER1	1
<i>CMOPT</i>	CMDEFV	0



## Déclaration RPG

```
D* MQCMHO Structure
D*
D*
D* Structure identifier
D  CMSID          1      4    INZ('CMHO')
D*
D* Structure version number
D  CMVER          5      8I 0  INZ(1)
D*
D* Options that control the action of MQCRTMH
D  CMOPT          9      12I 0 INZ(0)
```

## IBM i MQCNO (options de connexion) sous IBM i

La structure MQCNO permet à l'application de spécifier des options relatives à la connexion au gestionnaire de files d'attente local.

### Présentation

**Objectif:** La structure est un paramètre d'entrée/sortie dans l'appel MQCONNX.

**Versión:** la version actuelle de MQCNO est CNVER6. Les zones qui existent uniquement dans les versions les plus récentes de la structure sont identifiées comme telles dans les descriptions qui suivent.

Le fichier COPY fourni contient la version la plus récente de MQCNO prise en charge par l'environnement, mais avec la valeur initiale de la zone CNVER définie sur CNVER1. Pour utiliser des zones qui ne sont pas présentes dans la structure version-1, l'application doit définir la zone CNVER sur le numéro de version de la version requise.

**Jeu de caractères et codage:** les données de MQCNO doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par ENNAT.

- «Zones», à la page [1097](#)
- «Valeurs initiales», à la page [1103](#)
- «Déclaration RPG», à la page [1103](#)

### Zones

La structure MQCNO contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

#### CCDTUL (entier signé à 10 chiffres)

CCDTUL est la longueur de la chaîne identifiée par CCDTUP ou CCDTUU qui contient une URL qui identifie l'emplacement de la table de canaux de connexion client à utiliser pour la connexion.

Utilisez CCDTUL uniquement lorsque l'application émettant l'appel MQCONNX s'exécute en tant que IBM MQ MQI client.

Il s'agit d'une alternative par programmation à la définition des variables d'environnement [MQCHLLIB](#) et [MQCHLTAB](#).

Si l'application n'est pas exécutée en tant que client, CCDTUL est ignoré.

Cette zone est ignorée si CNVER est inférieur à CNVER6.

#### CCDTUU (entier signé à 10 chiffres)

CCDTUU est le décalage en octets, depuis le début de la structure MQCNO, vers une chaîne contenant une URL qui identifie l'emplacement de la table de canaux de connexion client à utiliser pour la connexion. Le décalage peut être positif ou négatif.

Utilisez CCDTUL uniquement lorsque l'application émettant l'appel MQCONNX s'exécute en tant que IBM MQ MQI client.

**Important :** Vous ne pouvez utiliser qu'une seule des deux CCDTUP et CCDTUO. L'appel échoue avec le code anomalie RC2600 si les deux zones sont différentes de zéro.

Il s'agit d'une alternative par programmation à la définition des variables d'environnement MQCHLLIB et MQCHLTAB .

Si l'application n'est pas exécutée en tant que client, CCDTUO est ignoré.

Cette zone est ignorée si CNVER est inférieur à CNVER6.

### **CCDTUP (pointeur)**

CCDTUP est un pointeur facultatif vers une chaîne qui contient une URL, afin d'identifier l'emplacement de la table de canaux de connexion client à utiliser pour la connexion.

Utilisez CCDTUP uniquement lorsque l'application émettant l'appel MQCONNX s'exécute en tant que IBM MQ MQI client.

**Important :** Vous ne pouvez utiliser qu'une seule des deux CCDTUP et CCDTUO. L'appel échoue avec le code anomalie RC2600 si les deux zones sont différentes de zéro.

Il s'agit d'une alternative par programmation à la définition des variables d'environnement MQCHLLIB et MQCHLTAB .

Si l'application n'est pas exécutée en tant que client, CCDTUP est ignoré.

Cette zone est ignorée si CNVER est inférieur à CNVER6.

### **CNAN (chaîne de caractères de 28 octets)**

Nom défini par l'application pour identifier la connexion au gestionnaire de files d'attente. La valeur initiale de la zone est des caractères nuls.

Cette zone est ignorée si CNVER est inférieur à CNVER7.

### **CNCCO (entier signé à 10 chiffres)**

Décalage en octets d'une structure de définition de canal MQCD à partir du début de la structure MQCNO.

### **CNCCP (pointeur)**

Il s'agit d'un pointeur vers une structure de définition de canal MQCD.

### **CNCONID (chaîne de caractères de 24 octets)**

Identificateur de connexion unique. Cette zone permet au gestionnaire de files d'attente d'identifier de manière fiable un processus d'application en lui affectant un identificateur unique lorsqu'il se connecte pour la première fois au gestionnaire de files d'attente.

Les applications utilisent l'identificateur de connexion à des fins de corrélation lors des appels PUT et GET. Le gestionnaire de files d'attente affecte un identificateur à toutes les connexions, quel que soit le mode d'établissement de la connexion.

Il est possible d'utiliser l'identificateur de connexion pour forcer la fin d'une unité d'oeuvre de longue durée. Pour ce faire, indiquez l'identificateur de connexion à l'aide de la commande PCF'Arrêter la connexion'ou de la commande MQSC STOP CONN. Pour plus d'informations sur l'utilisation de ces commandes, voir les liens associés.

La valeur initiale de la zone est de 24 octets nuls.

### **CNCT (chaîne de 128 octets)**

Il s'agit d'une balise que le gestionnaire de files d'attente associe aux ressources qui sont affectées par l'application lors de cette connexion.

Balise de connexion du gestionnaire de files d'attente.

Chaque application ou instance d'application doit utiliser une valeur différente pour la balise, afin que le gestionnaire de files d'attente puisse sérialiser correctement l'accès aux ressources affectées. Voir les descriptions des options CN\* CT\* pour plus de détails. La balise cesse d'être valide lorsque l'application s'arrête ou émet l'appel MQDISC.

Utilisez la valeur spéciale suivante si aucune balise n'est requise:

#### **CTAUCUN**

Aucune balise de connexion spécifiée.

La valeur est zéro binaire pour la longueur de la zone.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par LNCTAG. La valeur initiale de cette zone est CTNONE. Cette zone est ignorée si CNVER est inférieur à CNVER3.

Utilisez la zone ConnTag lors de la connexion à un gestionnaire de files d'attente z/OS .

#### **CNNORES2 (chaîne de caractères de 4 octets)**

Zone réservée pour remplir la structure jusqu'à une limite de 64 bits. La valeur initiale de la zone est zéro binaire pour la longueur de la zone.

Cette zone est ignorée si CNVER est inférieur à CNVER7.

#### **CNOPT (entier signé à 10 chiffres)**

Options qui contrôlent l'action de MQCONN.

#### **Options de liaison**

Les options de liaison contrôlent le type de liaison IBM MQ utilisé ; spécifiez une seule de ces options:

##### **CNSBND**

Liaison standard.

L'option de liaison standard permet à l'application et à l'agent du gestionnaire de files d'attente local de s'exécuter dans des unités d'exécution distinctes, généralement dans des processus distincts. L'agencement maintient l'intégrité du gestionnaire de files d'attente, c'est-à-dire qu'il protège le gestionnaire de files d'attente des programmes errants.

Utilisez CNSBND dans les situations où l'application n'a peut-être pas été entièrement testée ou n'est pas fiable ou n'est pas fiable. CNSBND est la valeur par défaut.

CNSBND est défini pour faciliter la documentation du programme. N'utilisez pas cette option avec une autre option contrôlant le type de liaison utilisé, mais comme sa valeur est égale à zéro, cette utilisation ne peut pas être détectée.

Cette option est prise en charge dans tous les environnements.

##### **CNFBND**

Liaison Fast Path.

L'option de liaison Fast Path permet à l'application et à l'agent du gestionnaire de files d'attente local de faire partie de la même unité d'exécution. Le raccourci contraste avec la liaison standard, où l'application et l'agent du gestionnaire de files d'attente local s'exécutent dans des unités d'exécution distinctes.

CNFBND est ignoré si le gestionnaire de files d'attente ne prend pas en charge ce type de liaison ; le traitement se poursuit comme si l'option n'avait pas été spécifiée.

CNFBND peut être avantageux dans les cas où plusieurs processus consomment plus de ressources que la ressource globale utilisée par l'application. Une application qui utilise la liaison Fast Path est appelée *application sécurisée*.

Tenez compte des points importants suivants lorsque vous décidez d'utiliser la liaison Fast Path:

- **L'utilisation de l'option CNFBND n'empêche pas une application de modifier ou d'altérer des messages et d'autres zones de données appartenant au gestionnaire de files d'attente. Utilisez cette option uniquement dans les situations où vous avez intégralement évalué ces problèmes.**
- L'application ne doit pas utiliser de signaux asynchrones ou d'interruptions de temporisateur (telles que sigkill) avec CNFBND. Il existe également des restrictions sur l'utilisation des segments de mémoire partagée.
- L'application ne doit pas avoir plus d'une unité d'exécution connectée au gestionnaire de files d'attente à la fois.
- L'application doit utiliser l'appel MQDISC pour se déconnecter du gestionnaire de files d'attente.
- L'application doit se terminer avant d'arrêter le gestionnaire de files d'attente à l'aide de la commande endmqm .

Les points suivants s'appliquent à l'utilisation de CNFBND dans les environnements indiqués:

- Sous IBM i, le travail doit s'exécuter sous le profil utilisateur QMQM qui appartient au groupe QMQMADM . En outre, le programme ne doit pas s'arrêter de manière anormale, sinon des résultats imprévisibles peuvent se produire.

Pour plus d'informations sur les implications de l'utilisation d'applications sécurisées, voir [Connexion à un gestionnaire de files d'attente à l'aide de l'appel MQCONN](#) et [Restrictions for trusted applications](#).

#### **CNSHBD**

Liaisons partagées.

L'option de liaisons partagées permet à l'application et à l'agent du gestionnaire de files d'attente local de s'exécuter dans des unités d'exécution distinctes, généralement dans des processus distincts. L'agencement maintient l'intégrité du gestionnaire de files d'attente, c'est-à-dire qu'il protège le gestionnaire de files d'attente des programmes errants. Toutefois, certaines ressources sont partagées entre l'application et l'agent du gestionnaire de files d'attente local. CNSHBD est ignoré si le gestionnaire de files d'attente ne prend pas en charge ce type de liaison. Le traitement se poursuit comme si l'option n'avait pas été spécifiée.

#### **CNIBND**

Liaisons isolées.

L'option de liaisons isolées permet à l'application et à l'agent du gestionnaire de files d'attente local de s'exécuter dans des unités d'exécution distinctes, généralement dans des processus distincts. L'agencement maintient l'intégrité du gestionnaire de files d'attente, c'est-à-dire qu'il protège le gestionnaire de files d'attente des programmes errants. Le processus d'application et l'agent de gestionnaire de files d'attente local sont isolés les uns des autres car ils ne partagent pas de ressources. CNIBND est ignoré si le gestionnaire de files d'attente ne prend pas en charge ce type de liaison. Le traitement se poursuit comme si l'option n'avait pas été spécifiée.

#### **Options de partage de descripteur**

Les options suivantes contrôlent le partage des descripteurs entre différentes unités d'exécution (unités de traitement parallèle) au sein d'un même processus. Une seule de ces options peut être spécifiée.

##### **CNHSN**

Aucun partage de descripteur entre les unités d'exécution.

L'option Pas de partage de descripteur entre les unités d'exécution indique que les descripteurs de connexion et d'objet ne peuvent être utilisés que par l'unité d'exécution qui a provoqué l'allocation du descripteur, c'est-à-dire l'unité d'exécution qui a émis l'appel MQCONN, MQCONN ou MQOPEN . Les descripteurs ne peuvent pas être utilisés par d'autres unités d'exécution appartenant au même processus.

## **CNHSB**

Partage des descripteurs de série entre les unités d'exécution, avec blocage des appels.

L'option de partage des descripteurs en série entre les unités d'exécution, avec blocage des appels, indique que les descripteurs de connexion et d'objet alloués par une unité d'exécution d'un processus peuvent être utilisés par d'autres unités d'exécution appartenant au même processus. Toutefois, une seule unité d'exécution à la fois peut utiliser une poignée particulière, c'est-à-dire que seule l'utilisation en série d'une poignée est autorisée. Si une unité d'exécution tente d'utiliser un descripteur déjà utilisé par une autre unité d'exécution, l'appel se bloque (attend) jusqu'à ce que le descripteur soit disponible.

## **CNHSNB**

Partage de descripteur de série entre les unités d'exécution, sans blocage d'appel.

L'option de partage de descripteur de série entre les unités d'exécution, sans blocage d'appel, est identique à l'option " *avec l'option blocking* ", sauf que si le descripteur est utilisé par une autre unité d'exécution, l'appel se termine immédiatement avec CCFAIL et RC2219 au lieu de se bloquer jusqu'à ce que le descripteur soit disponible.

Une unité d'exécution peut avoir zéro ou un descripteur non partagé, plus zéro ou plusieurs descripteurs partagés:

- Chaque appel MQCONN ou MQCONNX qui spécifie CNHSN renvoie un nouveau descripteur non partagé sur le premier appel, et le même descripteur non partagé sur les appels suivants (en supposant qu'il n'y ait pas d'appel MQDISC). Le code anomalie est RC2002 pour le deuxième appel et les appels ultérieurs.
- Chaque appel MQCONNX qui spécifie CNHSB ou CNHSNB renvoie un nouveau descripteur partagé à chaque appel.

Les descripteurs d'objet héritent des mêmes propriétés de partage que le descripteur de connexion spécifié dans l'appel MQOPEN qui a créé le descripteur d'objet. En outre, les unités de travail héritent des mêmes propriétés de partage que le descripteur de connexion utilisé pour démarrer l'unité de travail ; si l'unité de travail est démarrée dans une unité d'exécution à l'aide d'un descripteur partagé, l'unité de travail peut être mise à jour dans une autre unité d'exécution à l'aide du même descripteur.

Si vous ne spécifiez pas d'option de partage de descripteur, la valeur par défaut est déterminée par l'environnement:

- Dans l'environnement Microsoft Transaction Server (MTS), la valeur par défaut est la même que CNHSB.
- Dans d'autres environnements, la valeur par défaut est la même que CNHSN.

## **Options de reconnexion**

Les options de reconnexion déterminent si une connexion peut être reconnectée. Seules les connexions client sont reconnectables.

### **fichier CNRCDF**

L'option de reconnexion est résolue avec sa valeur par défaut. Si aucune valeur par défaut n'est définie, la valeur de cette option est DISABLED. La valeur de l'option est transmise au serveur et peut être interrogée par **PCF** et **MQSC**.

### **CNRC**

L'application peut être reconnectée à n'importe quel gestionnaire de files d'attente cohérent avec la valeur du paramètre MQCONNX **QMNAME**. Utilisez l'option CNRC uniquement s'il n'y a pas d'affinité entre l'application client et le gestionnaire de files d'attente avec lequel elle a initialement établi une connexion. La valeur de l'option est transmise au serveur et peut être interrogée par **PCF** et **MQSC**.

### **champ CNRCD**

L'application ne peut pas être reconnectée. La valeur de l'option n'est pas transmise au serveur.

**CNRCQM**

L'application ne peut être reconnectée qu'au gestionnaire de files d'attente auquel elle était initialement connectée. Utilisez cette valeur si un client peut être reconnecté, mais qu'il existe une affinité entre l'application client et le gestionnaire de files d'attente avec lequel il a établi une connexion à l'origine. Choisissez cette valeur si vous souhaitez qu'un client se reconnecte automatiquement à l'instance de secours d'un gestionnaire de files d'attente haute disponibilité. La valeur de l'option est transmise au serveur et peut être interrogée par **PCF** et **MQSC**.

Utilisez les options CNRC, CNRCDet CNRCQM uniquement pour les connexions client. Si les options sont utilisées pour une connexion de liaison, MQCONN échoue avec le code achèvement MQCC\_FAILED et le code anomalie MQRC\_OPTIONS\_ERROR.

**Option par défaut:** Si aucune des options décrites n'est requise, l'option suivante peut être utilisée:

**CNNONE**

Aucune option n'est spécifiée.

CNNONE est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette option soit utilisée avec une autre option CN\*, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

**CNSCO (entier signé à 10 chiffres)**

Décalage en octets d'une structure MQSCO à partir du début de la structure MQCNO.

Cette zone est ignorée si CNVER est inférieur à CNVER4.

**CNSCP (pointeur)**

Il s'agit de l'adresse d'une structure MQSCO.

Cette zone est ignorée si CNVER est inférieur à CNVER4.

**CNSECPO (entier signé à 10 chiffres)**

Décalage des paramètres de sécurité. Décalage de la structure MQCSP utilisée pour spécifier un ID utilisateur et un mot de passe.

La valeur peut être positive ou négative. La valeur initiale de cette zone est 0.

Cette zone est ignorée si CNVER est inférieur à CNVER5.

**CNSECPP (pointeur)**

Pointeur des paramètres de sécurité. Adresse de la structure MQCSP utilisée pour spécifier un ID utilisateur et un mot de passe.

La valeur initiale de cette zone est un pointeur NULL ou des octets NULL.

Cette zone est ignorée si CNVER est inférieur à CNVER5.

**CNSID (chaîne de caractères de 4 octets)**

Identificateur de la structure MQCNO.

La valeur doit être:

**NSIDV**

Identificateur de la structure des options de connexion.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est CNSIDV.

**CNVER (entier signé à 10 chiffres)**

Numéro de version de la structure MQCNO.

La valeur doit être:

## CNVER6

Structure d'options de connexion Version-6 .

Cette version est prise en charge dans tous les environnements.

## CNVER7

Structure des options de connexion Version-7 .

Cette version est prise en charge dans tous les environnements.

La constante suivante indique le numéro de version de la version en cours:

## CNVERC

Version actuelle de la structure des options de connexion.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est CNVER7.

## Valeurs initiales

Tableau 693. Zones dans MQCNO		
Nom de zone	Nom de la constante	Valeur de la constante
CNSID	NSIDV	' CNO↵
CNVER	CNVER5	1
CNOPT	CNAUCUN	0
CNCCO	Aucun	0
CNCCP	Aucun	Pointeur null ou octets null
CNCT	CTAUCUN	Valeurs NULL
CNSCP	Aucun	Pointeur null ou octets null
CNSCO	Aucun	0
CNCONID	Aucun	Valeurs NULL
CNSECPO	Aucun	0
CNSECPP	Aucun	Pointeur null ou octets null
CCDTUL	Aucun	0
CCDTUO	Aucun	0
CCDTUP	Aucun	Pointeur null ou octets null

### Remarques :

1. Le symbole ↵ représente un caractère blanc unique.

## Déclaration RPG

```
D*****  
D**  
D**          IBM MQ for IBM i          **  
D**          **                        **  
D** FILE NAME:      CMQCNOG           **  
D**          **                        **  
D** DESCRIPTION:    MQCNO Structure -- Connect Options **  
D**          **                        **  
D*****  
D** <N_OCO_COPYRIGHT>                **  
D** Licensed Materials - Property of IBM **  
D**          **                        **  
D** 5724-H72                          **  
D** (c) Copyright IBM Corp. 1993, 2024. All Rights Reserved. **
```

```

D**
D** US Government Users Restricted Rights - Use, duplication or
D** disclosure restricted by GSA ADP Schedule Contract with
D** IBM Corp.
D** <NOC_COPYRIGHT>
D*****
D**
D** FUNCTION:          This file declares the structure MQCNO,
D**                   which is used by the main MQI.
D**
D** PROCESSOR:        RPG (ILE)
D**
D*****
D*
D*
D*****
D** <BEGIN_BUILDINFO>
D** Generated on:    08/02/16 13:50
D** Build Level:    L000000
D** Build Type:    Production
D** Pointer Size:  128 Bit
D** Source File:
D** CMQCNOG
D** <END_BUILDINFO>
D*****
D*
D*.1....:....2....:....3....:....4....:....5....:....6....:....7..
D*
D*
D* MQCNO Structure
D*
D* Structure identifier
D  CNSID          1      4      INZ('CNO ')
D* Structure version number
D  CNVER          5      8I 0  INZ(1)
D* Options that control the action of MQCONN
D  CNOPT          9      12I 0 INZ(0)
D* Ver:1 **
D* Offset of MQCD structure for client connection
D  CNCCO          13     16I 0  INZ(0)
D* Address of MQCD structure for client connection
D  CNCCP          17     32*   INZ(*NULL)
D* Ver:2 **
D* Queue managerconnection tag
D  CNCT           33     160   INZ(X'0000000000000000-
D                               0000000000000000000000-
D                               0000000000000000000000-
D                               0000000000000000000000-
D                               0000000000000000000000-
D                               0000000000000000000000-
D                               0000000000000000000000-
D                               0000000000000000000000-
D                               000000000000')
D* Ver:3 **
D* Address of MQSCO structure for client connection
D  CNSCP          161    176*   INZ(*NULL)
D* Offset of MQSCO structure for client connection
D  CNSCO          177    180I 0 INZ(0)
D* Ver:4 **
D* Unique Connection Identifier
D  CNCONID        181    204   INZ(X'0000000000000000-
D                               0000000000000000000000-
D                               000000')
D* Offset of MQCSP structure
D  CNSECPO        205    208I 0 INZ(0)
D* Address of MQCSP structure
D  CNSECPP        209    224*   INZ(*NULL)
D* Ver:5 **
D* Address of CCDT URL string
D  CNCCDTUP       225    240*   INZ(*NULL)
D* Offset of CCDT URL string
D  CNCCDTUO       241    244I 0 INZ(0)
D* Length of CCDT URL
D  CNCCDTUL       245    248I 0 INZ(0)
D* Ver:6 **
D*
D*****
D** End of CMQCNOG
D*****

```



Récapitulatif de la structure MQCSP pour IBM i.

## Présentation

**Objectif:** La structure MQCSP permet au service d'autorisation d'authentifier un ID utilisateur et un mot de passe. Vous spécifiez la structure des paramètres de sécurité de connexion MQCSP sur un appel MQCONN.

**Jeu de caractères et codage:** les données dans MQCSP doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par ENNAT.

- «Zones», à la page [1105](#)
- «Valeurs initiales», à la page [1107](#)
- «Déclaration RPG», à la page [1108](#)

## Zones

La structure MQCSP contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

### **CSAUTH (entier signé à 10 chiffres)**

Il s'agit du type d'authentification à effectuer.

Les valeurs valides sont :

#### **Réseau de stockage SAN**

N'utilisez pas les zones d'ID utilisateur et de mot de passe.

#### **CSAUIAP**

Authentifiez les zones d'ID utilisateur et de mot de passe.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est CSAN.

### **CSCPPL (entier signé à 10 chiffres)**

Longueur du mot de passe à utiliser pour l'authentification.

La longueur maximale du mot de passe ne dépend pas de la plateforme. Si la longueur du mot de passe est supérieure à celle autorisée, la demande d'authentification échoue avec un RC2035.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

### **CSCPPO (entier signé à 10 chiffres)**

Il s'agit du décalage en octets du mot de passe à utiliser pour l'authentification.

Le décalage peut être positif ou négatif.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

### **CSCPPP (pointeur)**

Il s'agit de l'adresse du mot de passe à utiliser pour l'authentification.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null.

### **CSCSPUIL (entier signé à 10 chiffres)**

Longueur de l'ID utilisateur à utiliser pour l'authentification.

La longueur maximale de l'ID utilisateur ne dépend pas de la plateforme. Si la longueur de l'ID utilisateur est supérieure à celle autorisée, la demande d'authentification échoue avec un RC2035.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

### **CSCSPUIO (entier signé à 10 chiffres)**

Décalage en octets de l'ID utilisateur à utiliser pour l'authentification.

Le décalage peut être positif ou négatif.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

### **CSCSPUIP (pointeur)**

Il s'agit de l'adresse de l'ID utilisateur à utiliser pour l'authentification.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null. Cette zone est ignorée si CSVER est inférieur à CSVER5.

V 9.3.0

V 9.3.0

### **CSINITKL (entier signé à 10 chiffres)**

Longueur de la clé initiale pour le système de protection par mot de passe.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

V 9.3.0

V 9.3.0

### **CSINITKO (entier signé à 10 chiffres)**

Décalage en octets de la clé initiale pour le système de protection par mot de passe. Le décalage peut être positif ou négatif.

Vous pouvez utiliser *CSINITKO* ou *CSINITKP* pour spécifier la clé initiale, mais pas les deux. Pour plus d'informations, voir la description de la zone *CSINITKP*.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

V 9.3.0

V 9.3.0

### **CSINITKP (pointeur)**

Il s'agit de l'adresse en octets de la clé initiale pour le système de protection par mot de passe.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null.

IBM MQ MQI clients peut fournir la valeur de certaines zones sous forme de valeurs qui ont été chiffrées à l'aide du système de protection par mot de passe IBM MQ . Les zones suivantes peuvent contenir des valeurs chiffrées:

- Mot de passe du référentiel de clés, dans la structure MQSCO.

Une clé initiale est utilisée par l'algorithme de chiffrement pour chiffrer et déchiffrer ces valeurs. Si une clé initiale est fournie lorsque les valeurs de ces zones sont chiffrées à l'aide de l'utilitaire **runmqicred**, la même clé initiale doit être spécifiée par le client lorsqu'il se connecte au gestionnaire de files d'attente.

La clé initiale spécifiée à l'aide de cette zone remplace toute clé initiale spécifiée à l'aide de la variable d'environnement *MQS\_MQI\_KEYFILE* ou de la propriété *MQIInitialKeyFile* dans la section Sécurité du fichier de configuration du client.

Vous pouvez utiliser *CSINITKO* ou *CSINITKP* pour spécifier la clé initiale, mais pas les deux.

### **CSRE1 (chaîne de caractères de 4 octets)**

Zone réservée, requise pour l'alignement du pointeur sur IBM i.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est nulle.

### **CSRS2 (chaîne de caractères de 8 octets)**

Zone réservée, requise pour l'alignement du pointeur sur IBM i.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est nulle.

### **CSSID (chaîne de caractères de 4 octets)**

Identificateur de structure.

La valeur doit être:

**CSIDV**

Identificateur de la structure des paramètres de sécurité.

**CSVER (entier signé à 10 chiffres)**

Numéro de version de la structure.

La valeur doit être:

**CSVER1**

Structure des paramètres de sécurité Version-1 .



Structure des paramètres de sécurité Version-2 .







La constante suivante indique le numéro de version de la version en cours:

**CSVERC**

Version actuelle de la structure des paramètres de sécurité.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est CSVER1.

**Valeurs initiales**

<i>Tableau 694. Zones dans MQCNO</i>		
<b>Nom de zone</b>	<b>Nom de la constante</b>	<b>Valeur de la constante</b>
<i>CSSID</i>	CSIDV	'CSP↵'
<i>CSVER</i>	CSVER1	1
<i>CSAUTHT</i>	Aucun	0
<i>CSRE1</i>	Aucun	Valeurs NULL
<i>CSCSPUIP</i>	Aucun	Pointeur NULL
<i>CSCSPUIO</i>	Aucun	0
<i>CSCSPUIL</i>	Aucun	0
<i>CSRS2</i>	Aucun	Valeurs NULL
<i>CSCPPP</i>	Aucun	Pointeur NULL
<i>CSCPPO</i>	Aucun	0
<i>CSCPPL</i>	Aucun	0
  <i>CSINITKP</i>	Aucun	Pointeur NULL
  <i>CSINITKO</i>	Aucun	0
  <i>CSINITKL</i>	Aucun	0

**Remarque :**

1. Le symbole ↵ représente un caractère blanc unique.

## Déclaration RPG

```
D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQCSP Structure
D*
D* Structure identifier
D CSSID 1 4 INZ('CSP ')
D* Structure version number
D CSVER 5 8I 0 INZ(1)
D* Type of authentication
D CSAUTH 9 12I 0 INZ(0)
D* Reserved
D CSRE1 13 16 INZ(X'00000000')
D* Address of user ID
D CSCSPUIP 17 32* INZ(*NULL)
D* Offset of user ID
D CSCSPUIO 33 36I 0 INZ(0)
D* Length of user ID
D CSCSPUIL 37 40I 0 INZ(0)
D* Reserved
D CSRS2 41 48 INZ(X'0000000000000000')
D* Address of password
D CSCPPP 49 64* INZ(*NULL)
D* Offset of password
D CSCPP0 65 68I 0 INZ(0)
D* Length of password
D CSCPPL 69 72I 0 INZ(0)
```

## MQCTLO (structure des options de rappel de contrôle) sous IBM i

Structure spécifiant la fonction de rappel de contrôle.

### Présentation

#### Objet

La structure MQCTLO permet de spécifier des options relatives à une fonction de rappel de contrôle.

La structure est un paramètre d'entrée et de sortie dans l'appel [MQCTL](#).

#### Version

La version actuelle de MQCTLO est CTLV1.

#### Jeu de caractères et codage

Les données de MQCTLO doivent se trouver dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par ENNAT. Toutefois, si l'application s'exécute en tant que client IBM MQ, la structure doit être dans le jeu de caractères et le codage du client.

- «Zones», à la page [1108](#)
- «Valeurs initiales», à la page [1109](#)
- «Déclaration RPG», à la page [1110](#)

### Zones

La structure MQCTLO contient les zones suivantes ; les zones sont décrites par ordre alphabétique:

#### **COCONNAREA (entier signé à 10 chiffres)**

Structure des options de contrôle-Zone ConnectionArea .

Il s'agit d'une zone disponible que la fonction de rappel peut utiliser.

Le gestionnaire de files d'attente ne prend aucune décision en fonction du contenu de cette zone et est transmis tel quel à partir de la zone [CBCCONNAREA](#) de la structure MQCBC, qui est un paramètre de l'appel MQCB.

Cette zone est ignorée pour toutes les opérations autres que CTLSR et CTLSW.

Il s'agit d'une zone d'entrée et de sortie de la fonction de rappel. La valeur initiale de cette zone est un pointeur NULL ou des octets NULL.

### **COOPT (entier signé à 10 chiffres)**

Options qui contrôlent l'action de MQCTLO.

#### **CTLFQ**

Force l'échec de l'appel MQCTLO si le gestionnaire de files d'attente ou la connexion est à l'état de mise au repos.

Spécifiez GMFIQ, dans les options MQGMO transmises à l'appel MQCB, pour envoyer une notification aux consommateurs de message lorsqu'ils sont mis au repos.

#### **CTLTHR**

Cette option informe le système que l'application requiert que tous les consommateurs de messages, pour la même connexion, soient appelés sur la même unité d'exécution.

**Option par défaut:** si vous n'avez besoin d'aucune des options décrites, utilisez l'option suivante:

#### **CTLNO**

Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. Toutes les options sont définies sur leur valeur par défaut. CTLNO est défini pour aider à la documentation du programme ; il n'est pas prévu que cette option soit utilisée avec d'autres, mais comme sa valeur est zéro, une telle utilisation ne peut pas être détectée.

Il s'agit d'une zone d'entrée. La valeur initiale de la zone *COOPT* est CTLNO.

### **CORSV (entier signé à 10 chiffres)**

Il s'agit d'une zone réservée. La valeur initiale de cette zone est un caractère blanc.

### **COSID (entier signé à 10 chiffres)**

Structure des options de contrôle-Zone StrucId .

Il s'agit de l'identificateur de structure ; la valeur doit être:

#### **CTLSI**

Identificateur de la structure des options de contrôle.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est CTLSI.

### **COVER (entier signé à 10 chiffres)**

Structure des options de contrôle-Zone Version.

Il s'agit du numéro de version de la structure ; la valeur doit être:

#### **CTLV1**

Version-1 Structure des options de contrôle.

La constante suivante indique le numéro de version de la version en cours:

#### **CTLCV**

Version actuelle de la structure des options de contrôle.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est CTLV1.

## **Valeurs initiales**

<i>Tableau 695. Zones dans MQCTLO</i>		
<b>Nom de zone</b>	<b>Nom de la constante</b>	<b>Valeur de la constante</b>
<i>COSID</i>	CTLSI	'CTLO'
<i>COVER</i>	CTLV1	1
<i>COOPT</i>	CTLNO	Valeurs NULL

Tableau 695. Zones dans MQCTLO (suite)		
Nom de zone	Nom de la constante	Valeur de la constante
CORSV	Réservé, zone	
COCONNAREA	Aucun	Pointeur null ou octets null

## Déclaration RPG

```

D* MQCTLO Structure
D*
D*
D* Structure identifier
D COSID          1      4    INZ('CTLO')
D*
D* Structure version number
D COVER          5      8I 0 INZ(1)
D*
D* Options that control the action of MQCTL
D COOPT          9     12I 0 INZ(0)
D*
D* Reserved
D CORSV         13     16I 0 INZ(-1)
D*
D* MQCTL Data area passed to the function
D COCONNAREA    17     32*  INZ(*NULL)

```

## MQDH (en-tête de distribution) sur IBM i

La structure MQDH décrit les données supplémentaires présentes dans un message lorsque ce message est un message de liste de distribution stocké dans une file d'attente de transmission.

### Présentation

**Objectif:** Un message de liste de distribution est un message envoyé à plusieurs files d'attente de destination. Les données supplémentaires sont constituées de la structure MQDH suivie d'un tableau d'enregistrements MQOR et d'un tableau d'enregistrements MQPMR.

Cette structure est destinée à être utilisée par des applications spécialisées qui placent des messages directement dans des files d'attente de transmission ou qui suppriment des messages des files d'attente de transmission (par exemple, des agents de canal de message).

Cette structure ne doit pas être utilisée par les applications normales qui souhaitent simplement placer des messages dans des listes de distribution. Ces applications doivent utiliser la structure MQOD pour définir les destinations dans la liste de distribution et la structure MQPMO pour spécifier les propriétés de message ou recevoir des informations sur les messages envoyés aux destinations individuelles.

**Jeu de caractères et codage:** les données de MQDH doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par ENNAT pour le langage de programmation C.

Le jeu de caractères et le codage de la MQDH doivent être définis dans les zones *MDCSI* et *MDENC* dans:

- MQMD (si la structure MQDH est au début des données de message), ou
- Structure d'en-tête qui précède la structure MQDH (tous les autres cas).

**Utilisation:** Lorsqu'une application insère un message dans une liste de distribution et que certaines ou toutes les destinations sont distantes, le gestionnaire de files d'attente préfixe les données de message d'application avec les structures MQXQH et MQDH et place le message dans la file d'attente de transmission appropriée. Les données se trouvent donc dans l'ordre suivant lorsque le message se trouve dans une file d'attente de transmission:

- Structure MQXQH

- Structure MQDH plus tableaux d'enregistrements MQOR et MQPMR
- Données de message d'application

Selon les destinations, plusieurs messages de ce type peuvent être générés par le gestionnaire de files d'attente et placés dans des files d'attente de transmission différentes. Dans ce cas, les structures MQDH de ces messages identifient différents sous-ensembles des destinations définies par la liste de distribution ouverte par l'application.

Une application qui insère un message de liste de distribution directement dans une file d'attente de transmission doit se conformer à la séquence décrite précédemment et doit s'assurer que la structure MQDH est correcte. Si la structure MQDH n'est pas valide, le gestionnaire de files d'attente peut choisir de faire échouer l'appel MQPUT ou MQPUT1 avec le code anomalie RC2135.

Les messages peuvent être stockés dans une file d'attente sous forme de liste de distribution uniquement si la file d'attente est définie comme pouvant prendre en charge les messages de liste de distribution (voir l'attribut de file d'attente **DistLists** décrit dans «Attributs des files d'attente», à la page 1435 ). Si une application insère un message de liste de distribution directement dans une file d'attente qui ne prend pas en charge les listes de distribution, le gestionnaire de files d'attente divise le message de liste de distribution en messages individuels et place ces messages dans la file d'attente.

- «Zones», à la page 1111
- «Valeurs initiales», à la page 1114
- «Déclaration RPG», à la page 1114

## Zones

La structure MQDH contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

### DHCNT (entier signé à 10 chiffres)

Nombre d'enregistrements MQOR présents.

Définit le nombre de destinations. Une liste de distribution doit toujours contenir au moins une destination. Par conséquent, *DHCNT* doit toujours être supérieur à zéro.

La valeur initiale de cette zone est 0.

### DHCSI (entier signé à 10 chiffres)

Identificateur de jeu de caractères des données qui suivent les enregistrements MQOR et MQPMR.

Indique l'identificateur de jeu de caractères des données qui suivent les tableaux des enregistrements MQOR et MQPMR ; il ne s'applique pas aux données de type caractère dans la structure MQDH elle-même.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données. La valeur spéciale suivante peut être utilisée:

#### CSINHT

Hérite de l'identificateur de jeu de caractères de cette structure.

Les données de type caractères dans les données *suivant* cette structure se trouvent dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, la valeur CSINHT n'est pas renvoyée par l'appel MQGET.

CSINHT ne peut pas être utilisé si la valeur de la zone *MDPAT* dans MQMD est ATBRKR.

La valeur initiale de cette zone est CSUNDF.

### DHENC (entier signé à 10 chiffres)

Codage numérique des données qui suivent les enregistrements MQOR et MQPMR.

Indique le codage numérique des données qui suivent les tableaux des enregistrements MQOR et MQPMR ; il ne s'applique pas aux données numériques de la structure MQDH elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données.

La valeur initiale de cette zone est 0.

### **DHFLG (entier signé à 10 chiffres)**

Indicateurs généraux.

L'indicateur suivant peut être spécifié:

#### **DHFNOUVEAU**

Générez de nouveaux identificateurs de message.

Cet indicateur indique qu'un nouvel identificateur de message doit être généré pour chaque destination de la liste de distribution. Ce paramètre peut être défini uniquement lorsqu'aucun enregistrement d'insertion de message n'est présent ou lorsque les enregistrements sont présents mais qu'ils ne contiennent pas la zone *PRMID* .

L'utilisation de cet indicateur diffère la génération des identifiants de message jusqu'au dernier moment possible, à savoir le moment où le message de liste de distribution est finalement scindé en messages individuels. Cela réduit la quantité d'informations de contrôle qui doivent être transmises avec le message de liste de distribution.

Lorsqu'une application insère un message dans une liste de distribution, le gestionnaire de files d'attente définit DHFNEW dans la MQDH qu'elle génère lorsque les deux instructions suivantes sont vraies:

- Aucun enregistrement d'insertion de message n'est fourni par l'application ou les enregistrements fournis ne contiennent pas la zone *PRMID* .
- La zone *MDMID* de MQMD est MINONE ou la zone *PMOPT* de MQPMO inclut PMNMID

Si aucun indicateur n'est nécessaire, les éléments suivants peuvent être spécifiés:

#### **DHFNON**

Aucun indicateur.

Cette constante indique qu'aucun indicateur n'a été spécifié. DHFNON est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette constante soit utilisée avec une autre, mais comme sa valeur est nulle, une telle utilisation ne peut pas être détectée.

La valeur initiale de cette zone est DHFNON.

### **DHFMT (chaîne de caractères de 8 octets)**

Nom de format des données qui suivent les enregistrements MQOR et MQPMR.

Indique le nom de format des données qui suivent les tableaux des enregistrements MQOD et MQPMR (selon la dernière occurrence).

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de la zone *MDFMT* dans MQMD.

La valeur initiale de cette zone est FMNONE.

### **DHLEN (entier signé à 10 chiffres)**

Longueur de la structure MQDH plus les enregistrements MQOR et MQPMR suivants.

Il s'agit du nombre d'octets entre le début de la structure MQDH et le début des données de message suivant les tableaux d'enregistrements MQOR et MQPMR. Les données se produisent dans l'ordre suivant:

- Structure MQDH
- Tableau d'enregistrements MQOR



- Tableau d'enregistrements MQPMR
- Données de message

Les tableaux des enregistrements MQOR et MQPMR sont traités par des décalages contenus dans la structure MQDH. Si ces décalages génèrent des octets inutilisés entre une ou plusieurs structures MQDH, les tableaux d'enregistrements et les données de message, ces octets inutilisés doivent être inclus dans la valeur de *DHLEN*, mais le contenu de ces octets n'est pas conservé par le gestionnaire de files d'attente. Le tableau des enregistrements MQPMR doit précéder le tableau des enregistrements MQOR.

La valeur initiale de cette zone est 0.

### **DHORO (entier signé à 10 chiffres)**

Décalage du premier enregistrement MQOR à partir du début de MQDH.

Cette zone indique le décalage en octets du premier enregistrement du tableau des enregistrements d'objet MQOR contenant les noms des files d'attente de destination. Ce tableau contient *DHCNT* enregistrements. Ces enregistrements (plus les octets ignorés entre le premier enregistrement d'objet et la zone précédente) sont inclus dans la longueur indiquée par la zone *DHLEN*.

Une liste de distribution doit toujours contenir au moins une destination. Par conséquent, *DHORO* doit toujours être supérieur à zéro.

La valeur initiale de cette zone est 0.

### **DHPRF (entier signé à 10 chiffres)**

Indicateurs indiquant les zones MQPMR qui sont présentes.

Aucun ou plusieurs des indicateurs suivants peuvent être spécifiés:

#### **PFMID**

La zone d'identificateur de message est présente.

#### **ID de fonction**

La zone d'identificateur de corrélation est présente.

#### **ID de groupe de travail**

La zone d'identificateur de groupe est présente.

#### **PFFB**

Un champ de commentaires est présent.

#### **PFACC**

La zone de jeton de comptabilité est présente.

Si aucune zone MQPMR n'est présente, les éléments suivants peuvent être spécifiés:

#### **PFNONE**

Aucune zone d'enregistrement d'insertion de message n'est présente.

PFNONE est défini pour aider à la documentation du programme. Il n'est pas prévu que cette constante soit utilisée avec d'autres, mais comme sa valeur est nulle, une telle utilisation ne peut pas être détectée.

La valeur initiale de cette zone est PFNONE.

### **DHPRO (entier signé à 10 chiffres)**

Décalage du premier enregistrement MQPMR à partir du début de MQDH.

Cette zone indique le décalage en octets du premier enregistrement du tableau des enregistrements de message d'insertion MQPMR contenant les propriétés de message. S'il est présent, ce tableau contient *DHCNT* enregistrements. Ces enregistrements (plus les octets ignorés entre le premier enregistrement de message inséré et la zone précédente) sont inclus dans la longueur indiquée par la zone *DHLEN*.

Les enregistrements de message d'insertion sont facultatifs ; si aucun enregistrement n'est fourni, *DHPRO* est égal à zéro et *DHPRF* a la valeur PFNONE.

La valeur initiale de cette zone est 0.

### DHSID (chaîne de caractères de 4 octets)

Identificateur de structure.

La valeur doit être:

#### DHSIDV

Identificateur de la structure d'en-tête de distribution.

La valeur initiale de cette zone est DHSIDV.

### DHVER (entier signé à 10 chiffres)

Numéro de version de la structure.

La valeur doit être:

#### DHVER1

Numéro de version de la structure d'en-tête de distribution.

La constante suivante indique le numéro de version de la version en cours:

#### Instruction DHVERC

Version actuelle de la structure d'en-tête de distribution.

La valeur initiale de cette zone est DHVER1.

## Valeurs initiales

Tableau 696. Zones dans MQDH		
Nom de zone	Nom de la constante	Valeur de la constante
DHSID	DHSIDV	'DH↵↵'
DHVER	DHVER1	1
DHLEN	Aucun	0
DHENC	Aucun	0
DHCSI	CSUNDF	0
DHFMT	FMAUCUN	Espaces vides
DHFLG	DHFNON	0
DHPRF	PFNONE	0
DHCNT	Aucun	0
DHORO	Aucun	0
DHPRO	Aucun	0

### Remarques :

1. Le symbole ↵ représente un caractère blanc unique.

## Déclaration RPG

```

D*..1.....2.....3.....4.....5.....6.....7..
D* MQDH Structure
D*
D* Structure identifier
D  DHSID           1      4  INZ('DH ')
D* Structure version number
D  DHVER           5      8I 0 INZ(1)

```

```

D* Length of MQDH structure plus following MQOR and MQPMR records
D DHLEN          9      12I 0 INZ(0)
D* Numeric encoding of data that follows the MQOR and MQPMR records
D DHENC          13     16I 0 INZ(0)
D* Character set identifier of data that follows the MQOR and MQPMR
D* records
D DHCSI          17     20I 0 INZ(0)
D* Format name of data that follows the MQOR and MQPMR records
D DHFMT          21     28    INZ(' ')
D* General flags
D DHFLG          29     32I 0 INZ(0)
D* Flags indicating which MQPMR fields are present
D DHPRF          33     36I 0 INZ(0)
D* Number of MQOR records present
D DHCNT          37     40I 0 INZ(0)
D* Offset of first MQOR record from start of MQDH
D DHORO          41     44I 0 INZ(0)
D* Offset of first MQPMR record from start of MQDH
D DHPRO          45     48I 0 INZ(0)

```

## IBM i MQDLH (en-tête de rebut) sur IBM i

### Présentation

#### Objet

La structure MQDLH décrit les informations qui préfixent les données de message d'application des messages de la file d'attente des messages non livrés. Un message peut arriver dans la file d'attente de rebut car le gestionnaire de files d'attente ou l'agent de canal de message l'a redirigé vers la file d'attente. Une application peut placer le message directement dans la file d'attente.

#### Format

FMDLH

#### Jeu de caractères et codage

Le MQDLH peut être au début des données de message d'application. Si tel est le cas, les zones de la structure MQDLH se trouvent dans le jeu de caractères et le codage donnés par les zones MDCSI et MDENC. Si ce n'est pas le cas, le jeu de caractères et le codage sont définis par les zones MDCSI et MDENC dans la structure d'en-tête qui précède MQDLH.

Le jeu de caractères doit comporter des caractères mono-octet pour les caractères admis dans les noms de file d'attente.

#### Utilisation

Les applications qui placent des messages directement dans la file d'attente de rebut doivent ajouter une structure MQDLH aux données de message et initialiser les zones avec les valeurs appropriées. Toutefois, le gestionnaire de files d'attente ne requiert pas qu'une structure MQDLH soit présente ou que des valeurs valides soient spécifiées pour les zones.

Si un message est trop long pour être inséré dans la file d'attente de rebut, l'application doit envisager d'effectuer l'une des opérations suivantes:

- Tronque les données de message pour qu'elles tiennent dans la file d'attente des messages non livrés.
- Enregistrez le message dans la mémoire secondaire et placez un message de rapport d'exception dans la file d'attente des messages non livrés pour indiquer que le message est trop long.
- Supprimez le message et renvoyez une erreur à son émetteur. Si le message est un message critique. Ne supprimez le message que s'il est connu que l'émetteur possède toujours une copie du message. Par exemple, un message reçu par un agent MCA à partir d'un canal de communication.

Les choix appropriés dépendent de la conception de l'application.

Le gestionnaire de files d'attente effectue un traitement spécial lorsqu'un message qui est un segment est inséré avec une structure MQDLH à l'avant. Pour plus de détails, voir la description de la structure MQMDE.

- [«Insertion de messages dans la file d'attente de rebut», à la page 1116](#)

- «Obtention de messages de la file d'attente des messages non livrés», à la page [1116](#)
- «Zones», à la page [1116](#)
- «Valeurs initiales», à la page [1120](#)
- «Déclaration RPG», à la page [1121](#)

## Insertion de messages dans la file d'attente de rebut

Si un message est inséré dans la file d'attente de rebut, la structure MQMD utilisée pour l'appel MQPUT ou MQPUT1 doit être identique à la structure MQMD associée au message. MQMD est généralement celui renvoyé par l'appel MQGET , sauf dans les cas suivants:

- Les zones MDCSI et MDENC doivent être définies sur le jeu de caractères et le codage utilisés pour les zones de la structure MQDLH .
- La zone MDFMT doit être définie sur FMDLH pour indiquer que les données commencent par une structure MQDLH .
- Les zones de contexte, MDACC, MDAID, MDAOD, MDPAN, MDPAT, MDPD, MDPTet MDUID doivent être définies à l'aide d'une option de contexte adaptée aux circonstances:
  - Une application qui place dans la file d'attente de rebut un message qui n'est lié à aucun message précédent doit utiliser l'option PMDEFC . L'option PMDEFC permet au gestionnaire de files d'attente de définir toutes les zones de contexte du descripteur de message sur leurs valeurs par défaut.
  - Une application serveur plaçant dans la file d'attente de rebut un message qu'elle a reçu doit utiliser l'option PMPASA afin de conserver les informations de contexte d'origine.
  - Une application serveur plaçant dans la file d'attente des messages non livrés une réponse au message qu'elle a reçu doit utiliser l'option PMPASI . L'option PMPASI conserve les informations d'identité mais définit les informations d'origine comme étant celles de l'application serveur.
  - Un agent MCA qui place dans la file d'attente de rebut un message qu'il a reçu de son canal de communication doit utiliser l'option PMSETA . L'option PMSETA conserve les informations de contexte d'origine.

Dans la structure MQDLH elle-même, les zones doivent être définies comme suit:

- Les zones DLCSI, DLENCet *DLFMT* doivent être définies sur les valeurs qui décrivent les données qui suivent la structure MQDLH . Ces valeurs sont généralement celles du descripteur de message d'origine.
- Les zones de contexte DLPAT, DLPAN, DLPDet DLPT doivent être définies sur les valeurs appropriées à l'application qui place le message dans la file d'attente de rebut. Ces valeurs ne sont pas liées au message d'origine.
- D'autres zones doivent être définies selon les besoins.

L'application doit s'assurer que toutes les zones ont des valeurs valides et que les zones alphanumériques sont remplies avec des blancs à la longueur définie de la zone. Les données de type caractère ne doivent pas être arrêtées prématurément à l'aide d'un caractère NULL. Le gestionnaire de files d'attente ne convertit pas les caractères null et suivants en blancs dans la structure MQDLH .

## Obtention de messages de la file d'attente des messages non livrés

Les applications qui extraient des messages de la file d'attente de rebut doivent vérifier que les messages commencent par une structure MQDLH . L'application peut déterminer si une structure MQDLH est présente en examinant la zone MDFMT dans le descripteur de message MQMD. Si la zone a la valeur FMDLH, les données de message commencent par une structure MQDLH . Les messages de la file d'attente de rebut peuvent être tronqués s'ils étaient trop longs à l'origine pour la file d'attente à laquelle ils étaient destinés.

## Zones

La structure MQDLH contient les zones suivantes ; les zones sont décrites par ordre alphabétique:

### **DLCSI (entier signé à 10 chiffres)**

Identificateur de jeu de caractères des données qui suivent MQDLH.

DLCSI indique l'identificateur de jeu de caractères des données qui suivent la structure MQDLH . Les données proviennent généralement du message d'origine. Elle ne s'applique pas aux données de type caractères de la structure MQDLH elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. La valeur spéciale suivante peut être utilisée:

#### **CSINHT**

Hérite de l'identificateur de jeu de caractères de cette structure.

Les données de type caractères des données qui suivent cette structure sont dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, la valeur CSINHT n'est pas renvoyée par l'appel MQGET .

CSINHT ne peut pas être utilisé si la valeur de la zone MDPAT dans MQMD est ATBRKR.

La valeur initiale de cette zone est CSUNDF.

### **DLDM (chaîne de caractères de 48 octets)**

Nom du gestionnaire de files d'attente de destination d'origine.

Il s'agit du nom du gestionnaire de files d'attente qui était la destination d'origine du message.

La longueur de cette zone est indiquée par LNQMN. La valeur initiale de cette zone est de 48 caractères blancs.

### **DLDQ (chaîne de caractères de 48 octets)**

Nom de la file d'attente de destination d'origine.

Il s'agit du nom de la file d'attente de messages qui était la destination d'origine du message.

La longueur de cette zone est indiquée par LNQN. La valeur initiale de cette zone est de 48 caractères blancs.

### **DLENC (entier signé à 10 chiffres)**

Codage numérique des données qui suit MQDLH.

DLENC indique le codage numérique des données qui suivent la structure MQDLH . Les données proviennent généralement du message d'origine. Elle ne s'applique pas aux données numériques de la structure MQDLH elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données.

La valeur initiale de cette zone est 0.

### **DLFMT (chaîne de caractères de 8 octets)**

Nom de format des données qui suivent MQDLH.

Indique le nom de format des données qui suivent la structure MQDLH (généralement les données du message d'origine).

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de la zone MDFMT dans MQMD.

La longueur de cette zone est indiquée par LNFMT. La valeur initiale de cette zone est FMNONE.

### **DLPAN (chaîne de caractères de 28 octets)**

Nom de l'application qui a inséré le message dans la file d'attente de rebut (message non distribué).

Le format du nom dépend de la zone DLPAT . Voir la description de la zone MDPAN dans [«MQMD \(descripteur de message\) sous IBM i»](#), à la page 1162.

S'il s'agit du gestionnaire de files d'attente qui redirige le message vers la file d'attente de rebut, DLPAN contient les 28 premiers caractères du nom du gestionnaire de files d'attente. Le nom est complété par des blancs si nécessaire.

La longueur de cette zone est indiquée par LNPNAN. La valeur initiale de cette zone est de 28 caractères blancs.

### **DLPAT (entier signé à 10 chiffres)**

Type d'application qui a inséré le message dans la file d'attente de rebut (message non distribué).

Cette zone a la même signification que la zone MDPAT dans le descripteur de message MQMD (voir [«MQMD \(descripteur de message\) sous IBM i»](#), à la page 1162 pour plus de détails).

Si c'est le gestionnaire de files d'attente qui redirige le message vers la file d'attente de rebut, DLPAT a la valeur ATQM.

La valeur initiale de cette zone est 0.

### **DLPD (chaîne de caractères de 8 octets)**

Date à laquelle le message a été inséré dans la file d'attente de rebut (message non distribué).

Le format utilisé pour la date à laquelle cette zone est générée par le gestionnaire de files d'attente est le suivant:

- YYYYMMDD

où les caractères représentent:

#### **YYYY**

année (quatre chiffres)

#### **MM**

mois de l'année (01 à 12)

#### **DD**

jour du mois (01 à 31)

Le temps moyen de Greenwich (GMT) est utilisé pour les zones DLPD et DLPT , à condition que l'horloge système soit définie avec précision sur le temps moyen de Greenwich.

La longueur de cette zone est indiquée par LNPDAT. La valeur initiale de cette zone est de huit caractères blancs.

### **DLPT (chaîne de caractères de 8 octets)**

Heure à laquelle le message a été inséré dans la file d'attente de rebut (message non distribué).

Le format utilisé pour l'heure à laquelle cette zone est générée par le gestionnaire de files d'attente est le suivant:

- HHMMSSSTH

où les caractères représentent (dans l'ordre):

#### **HH**

heures (00 à 23)

#### **MM**

minutes (00 à 59)

#### **SS**

secondes (00 à 59 ; voir la remarque plus loin dans cette rubrique)

#### **T**

dixièmes de seconde (0 à 9)

## H

centièmes de seconde (0 à 9)

**Remarque :** Si l'horloge système est synchronisée selon une norme d'heure précise, il est possible que 60 ou 61 secondes soient renvoyées dans DLPT. La seconde supplémentaire se produit lorsque des secondes intercalaires sont insérées dans la norme de temps globale.

Le temps moyen de Greenwich (GMT) est utilisé pour les zones DLPD et DLPT, à condition que l'horloge système soit définie avec précision sur le temps moyen de Greenwich.

La longueur de cette zone est indiquée par LNPTIM. La valeur initiale de cette zone est de huit caractères blancs.

### **DLREA (entier signé à 10 chiffres)**

Le message anomalie est arrivé dans la file d'attente de rebut (message non distribué).

Identifie la raison pour laquelle le message a été placé dans la file d'attente de rebut au lieu de la file d'attente de destination d'origine. Il doit s'agir de l'une des valeurs FB\* ou RC\* (par exemple, RC2053). Voir la description de la zone *MDFB* dans «MQMD (descripteur de message) sous IBM i», à la page 1162 pour plus de détails sur les valeurs FB\* communes qui peuvent se produire.

Si la valeur est comprise entre FBIFST et FBILST, le code d'erreur IMS réel peut être déterminé en soustrayant FBIERR de la valeur de la zone *DLREA*.

Certaines valeurs FB\* apparaissent uniquement dans cette zone. Ils concernent des messages de référentiel, des messages de déclenchement ou des messages de file d'attente de transmission qui sont transférés dans la file d'attente de rebut. Ces valeurs sont les suivantes:

#### **FBABEG**

L'application ne peut pas être démarrée.

Une application traitant un message de déclencheur n'a pas pu démarrer l'application nommée dans la zone TMAI du message de déclencheur ; voir «MQTM-Message de déclenchement», à la page 1296.

#### **FBATYP**

Erreur de type d'application.

Une application traitant un message de déclenchement n'a pas pu démarrer l'application car la zone TMAI du message de déclenchement n'est pas valide ; voir «MQTM-Message de déclenchement», à la page 1296.

#### **FBOCD**

Canal récepteur de cluster supprimé.

Le message se trouvait dans une file d'attente de transmission de cluster destinée à une file d'attente de cluster qui a été ouverte avec l'option FBIERR. Le canal récepteur de cluster éloigné à utiliser pour transmettre le message à la file d'attente de destination a été supprimé avant l'envoi du message. Etant donné que FBIERR a été spécifié, seul le canal sélectionné lors de l'ouverture de la file d'attente peut être utilisé pour transmettre le message. Ce canal n'étant plus disponible, le message a été placé dans la file d'attente des messages non livrés.

#### **FBNARM**

Le message n'est pas un message de référentiel.

#### **FBSBCX**

Message arrêté par l'exit de définition automatique de canal.

#### **FBSBMX**

Message arrêté par l'exit de message de canal.

#### **FBTM**

Structure MQTM non valide ou manquante.

La zone MDFMT dans MQMD indique FMTM, mais le message ne commence pas par une structure MQTM valide. Par exemple, l'identificateur mnémorique *TMSID* peut ne pas être valide. Il se peut

que *TMVER* ne soit pas reconnu. La longueur du message de déclenchement peut être insuffisante pour contenir la structure MQTM .

#### **FBXQME**

Le format du message dans la file d'attente de transmission est incorrect.

Un agent MCA a détecté qu'un message de la file d'attente de transmission n'est pas au format correct. L'agent MCA place le message dans la file d'attente de rebut à l'aide de ce code retour.

La valeur initiale de cette zone est RCNONE.

#### **DLSID (chaîne de caractères de 4 octets)**

Identificateur de structure.

La valeur doit être:

#### **DLSIDV**

Identificateur de la structure d'en-tête de rebut.

La valeur initiale de cette zone est DLSIDV.

#### **DLVER (entier signé à 10 chiffres)**

Numéro de version de la structure.

La valeur doit être:

#### **DLVER1**

Numéro de version de la structure d'en-tête de rebut.

La constante suivante indique le numéro de version de la version en cours:

#### **DLVERC**

Version actuelle de la structure d'en-tête de rebut.

La valeur initiale de cette zone est DLVER1.

### **Valeurs initiales**

<i>Tableau 697. Champs dans MQDLH</i>		
<b>Nom de zone</b>	<b>Nom de la constante</b>	<b>Valeur de la constante</b>
DLSID	DLSIDV	'DLH~'
DLVER	DLVER1	1
DLREA	RCNONE	0
DLDQ	Aucun	Espaces vides
DLDM	Aucun	Espaces vides
DLENC	Aucun	0
DLCSI	CSUNDF	0
DLFMT	FMNONE	Espaces vides
DLPAT	Aucun	0
DLPAN	Aucun	Espaces vides
DLPD	Aucun	Espaces vides
DLPT	Aucun	Espaces vides

#### **Remarques :**

1. Le symbole ~ représente un caractère blanc unique.



## Déclaration RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D* MQLH Structure
D*
D* Structure identifier
D  DLSID          1          4      INZ('DLH ')
D* Structure version number
D  DLVER          5          8I 0 INZ(1)
D* Reason message arrived on dead-letter(undelivered-message) queue
D  DLREA          9          12I 0 INZ(0)
D* Name of original destination queue
D  DLDQ           13         60      INZ
D* Name of original destination queue manager
D  DLDM           61         108     INZ
D* Numeric encoding of data that followsMQLH
D  DLENC          109        112I 0 INZ(0)
D* Character set identifier of data thatfollows MQLH
D  DLCSI          113        116I 0 INZ(0)
D* Format name of data that followsMQLH
D  DLFMT          117        124     INZ(' ')
D* Type of application that put messageon dead-letter
D* (undelivered-message)queue
D  DLPAT          125        128I 0 INZ(0)
D* Name of application that put messageon dead-letter
D* (undelivered-message)queue
D  DLPAN          129        156     INZ
D* Date when message was put ondead-letter (undelivered-message)queue
D  DLPD           157        164     INZ
D* Time when message was put on thedead-letter (undelivered-message)queue
D  DLPT           165        172     INZ
```

IBM i

## MQDMHO (Delete message handle options) sous IBM i

La structure **MQDMHO** permet aux applications de spécifier des options qui contrôlent la façon dont les descripteurs de message sont supprimés.

### Présentation

**Objectif:** la structure est un paramètre d'entrée dans l'appel **MQDLTMH**.

**Jeu de caractères et codage:** les données dans **MQDMHO** doivent être dans le jeu de caractères de l'application et le codage de l'application (ENNAT).

- «Zones», à la page [1121](#)
- «Valeurs initiales», à la page [1122](#)
- «Déclaration RPG», à la page [1122](#)

### Zones

La structure **MQDMHO** contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

#### **DMOPT (entier signé à 10 chiffres)**

La valeur doit être:

**DMAUCUN**

Aucune option spécifiée.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **DMNONE**.

#### **DMSID (entier signé à 10 chiffres)**

Il s'agit de l'identificateur de structure ; la valeur doit être:

## DMSIDV

Identificateur de la structure des options de descripteur de message de suppression.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **DMSIDV**.

## DMVER (entier signé à 10 chiffres)

Il s'agit du numéro de version de la structure ; la valeur doit être:

### DMVER1

Version-1 : supprime la structure d'options de descripteur de message.

La constante suivante indique le numéro de version de la version en cours:

### DMVERC

Version actuelle de la structure d'options de suppression de descripteur de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **DMVER1**.

## Valeurs initiales

Nom de zone	Nom de la constante	Valeur de la constante
DMSID	DMSIDV	' DMHO '
DMVER	DMVER1	1
DMOPT	DMAUCUN	0

## Déclaration RPG

```
D* MQDMHO Structure
D*
D*
D* Structure identifier
D DMSID          1      4  INZ('DMHO')
D*
D* Structure version number
D DMVER          5      8I 0 INZ(1)
D*
D* Options that control the action of MQDLTMH
D DMOPT          9      12I 0 INZ(0)
```

## IBM i MQDMPO (Delete message property options) sous IBM i

Structure définissant les options de propriété de message de suppression.

## Présentation

**Objectif:** La structure MQDMPO permet aux applications de spécifier des options qui contrôlent la façon dont les propriétés des messages sont supprimées. La structure est un paramètre d'entrée dans l'appel MQDLTMP.

**Jeu de caractères et codage:** les données de MQDMPO doivent être dans le jeu de caractères de l'application et le codage de l'application (ENNAT).

- «Zones», à la page [1123](#)
- «Valeurs initiales», à la page [1124](#)
- «Déclaration RPG», à la page [1124](#)

## Zones

La structure MQDMPO contient les zones suivantes ; les zones sont décrites par ordre alphabétique:

### **DPOPT (entier signé à 10 chiffres)**

Supprimer la structure des options de propriété de message-Zone DPOPT.

**Options d'emplacement:** Les options suivantes sont liées à l'emplacement relatif de la propriété par rapport au curseur de propriété.

#### **DPDELF**

Supprime la première propriété qui correspond au nom spécifié.

#### **DPDELC**

Supprime la propriété pointée par le curseur de propriété ; il s'agit de la propriété qui a été interrogée pour la dernière fois à l'aide de l'option IPINQF ou IPINQN.

Le curseur de propriété est réinitialisé lorsque le descripteur de message est réutilisé. Il est également réinitialisé lorsque le descripteur de message est spécifié dans la zone *HMSG* du MQGMO sur un appel MQGET ou dans la structure MQPMO sur un appel MQPUT.

Le curseur de propriété est réinitialisé lorsque le descripteur de message est réutilisé ou lorsque le descripteur de message est spécifié dans la zone *HMSG* de la structure MQGMO sur une structure MQGET sur un appel MQGET ou dans une structure MQPMO sur un appel MQPUT.

L'appel échoue avec le code achèvement CCFAIL et la raison RC2471 si cette option est utilisée alors que le curseur de propriété n'a pas encore été établi. Elle échoue également avec ces codes si la propriété pointée par le curseur de propriété a déjà été supprimée.

Si aucune de ces options n'est requise, l'option suivante peut être utilisée:

#### **DPNONE**

Aucune option spécifiée.

La valeur initiale de cette zone d'entrée est DPDELF.

### **DPSID (entier signé à 10 chiffres)**

Supprimer la structure des options de propriété de message-Zone DPSID.

Il s'agit de l'identificateur de structure. La valeur doit être:

#### **DPSIDV**

Identificateur de la structure des options de suppression des propriétés de message.

Cette zone est toujours une zone d'entrée. La valeur initiale de cette zone est DPSIDV.

### **DPVER (entier signé à 10 chiffres)**

Supprimer la structure des options de propriété de message-Zone DPVER.

Il s'agit du numéro de version de la structure. La valeur doit être:

#### **DPVER1**

Numéro de version de la structure des options de suppression de propriété de message.

La constante suivante indique le numéro de version de la version en cours:

#### **DPVERC**

Version actuelle de la structure des options de suppression des propriétés de message.

Cette zone est toujours une zone d'entrée. La valeur initiale de cette zone est DPVER1.

## Valeurs initiales

Tableau 699. Zones dans MQDPMO		
Nom de zone	Nom de la constante	Valeur de la constante
DPSID	DPSIDV	'DMPO'
DPVER	DPVER1	1
DPOPT	Options qui contrôlent l'action de MQDLTMP	DPNONE

## Déclaration RPG

```
D* MQDPMO Structure
D*
D*
D* Structure identifier
D  DPSID          1      4    INZ('DMPO')
D*
D* Structure version number
D  DPVER          5      8I 0 INZ(1)
D*
** Options that control the action of
D* MQDLTMP
D  DPOPT          9      12I 0 INZ(0)
```

## IBM i MQEPH (en-tête PCF intégré) sous IBM i

### Présentation

#### Objet

La structure MQEPH décrit les données supplémentaires présentes dans un message lorsqu'il s'agit d'un message PCF (Programmable Command Format). La zone *EPPFH* définit les paramètres PCF qui suivent cette structure et permet de suivre les données de message PCF avec d'autres en-têtes.

#### Format

EPFMT

#### Jeu de caractères et codage

Les données de MQEPH doivent être dans le jeu de caractères et le codage du gestionnaire de files d'attente local ; cet attribut est fourni par l'attribut de gestionnaire de files d'attente **CCSID**.

Définissez le jeu de caractères et le codage du MQEPH dans les zones *MDCSI* et *MDENC* dans:

- Le MQMD (si la structure MQEPH est au début des données de message), ou
- Structure d'en-tête qui précède la structure MQEPH (tous les autres cas).

#### Utilisation

Vous ne pouvez pas utiliser les structures MQEPH pour envoyer des commandes au serveur de commandes ou à tout autre serveur acceptant les PCF du gestionnaire de files d'attente.

De même, le serveur de commandes ou tout autre serveur d'acceptation PCF du gestionnaire de files d'attente ne génère pas de réponses ou d'événements contenant des structures MQEPH.

- [«Zones», à la page 1124](#)
- [«Valeurs initiales», à la page 1126](#)
- [«Déclaration RPG», à la page 1127](#)

### Zones

La structure MQEPH contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

### **EPCSI (entier signé à 10 chiffres)**

Il s'agit de l'identificateur de jeu de caractères des données qui suivent la structure MQEPH et les paramètres PCF associés ; il ne s'applique pas aux données de type caractère de la structure MQEPH elle-même.

La valeur initiale de cette zone est EPCUND.

### **EPENC (entier signé à 10 chiffres)**

Il s'agit du codage numérique des données qui suit la structure MQEPH et les paramètres PCF associés ; il ne s'applique pas aux données de type caractères de la structure MQEPH elle-même.

La valeur initiale de cette zone est 0.

### **EPFLG (entier signé à 10 chiffres)**

Les valeurs suivantes sont disponibles :

#### **EPAUCUN**

Aucun indicateur n'a été spécifié. *MDCSI* EPNONE est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette constante soit utilisée avec d'autres, mais comme sa valeur est nulle, une telle utilisation ne peut pas être détectée.

#### **EPCSEM**

Le jeu de caractères des paramètres contenant des données de type caractère est spécifié individuellement dans la zone *CCSID* de chaque structure. Le jeu de caractères des zones *EPSID* et *EPFMT* est défini par *CCSID* dans la structure d'en-tête qui précède la structure MQEPH, ou par la zone *MDCSI* dans le MQMD si le MQEPH est au début du message.

La valeur initiale de cette zone est EPNONE.

### **EPFMT (chaîne de caractères de 8 octets)**

Il s'agit du nom de format des données qui suivent la structure MQEPH et les paramètres PCF associés.

La valeur initiale de cette zone est EPFMNO.

### **EPLEN (entier signé à 10 chiffres)**

Il s'agit de la quantité de données précédant la structure d'en-tête suivante. Il comprend :

- Longueur de l'en-tête MQEPH
- Longueur de tous les paramètres PCF suivant l'en-tête
- Tout remplissage vide suivant ces paramètres

EPLEN doit être un multiple de 4.

La partie de longueur fixe de la structure est définie par EPSTLF.

La valeur initiale de cette zone est 68.

### **EPPCFH (MQCFH)**

Il s'agit de l'en-tête PCF (Programmable Command Format) qui définit les paramètres PCF qui suivent la structure MQEPH. Cela vous permet de suivre les données de message PCF avec d'autres en-têtes.

L'en-tête PCF est initialement défini avec les valeurs suivantes:

Tableau 700. Champs de l'EPPCFH		
Nom de zone	Nom de la constante	Valeur de la constante
EP3TYP	CFTNON	0
EP3LEN	ENV FHLF	36
EP3VER	FHVER3	3

<i>Tableau 700. Champs de l'EPPCFH (suite)</i>		
<b>Nom de zone</b>	<b>Nom de la constante</b>	<b>Valeur de la constante</b>
EP3CMD	CMNONE	0
EP3SEQ	Aucun	1
EP3CTL	CFCLST	1
EEP3CC	CCOK	0
EP3REA	RCNONE	0
EP3CNT	Aucun	0

L'application doit remplacer EP3TYP par un type de structure valide pour l'utilisation de l'en-tête PCF imbriqué.

### **EPSID (chaîne de caractères de 4 octets)**

La valeur doit être:

#### **ID EPSTID**

Identificateur de la structure d'en-tête PCF intégrée.

La valeur initiale de cette zone est EPSTID.

### **EPVER (entier signé à 10 chiffres)**

La valeur peut être :

#### **EPVER1**

Numéro de version de la structure d'en-tête PCF intégrée.

La constante suivante indique le numéro de version de la version en cours:

#### **EPVER3**

Version actuelle de la structure d'en-tête PCF intégrée.

La valeur initiale de cette zone est EPVER3.

## **Valeurs initiales**

<i>Tableau 701. Zones dans MQEPH</i>		
<b>Nom de zone</b>	<b>Nom de la constante</b>	<b>Valeur de la constante</b>
EPSID	ID EPSTID	'EP↵↵'
EPVER	EPVER1	1
EPLEN	Fonction EPSTLF	68
EPENC	Aucun	0
EPCSI	EPCUND	0
EPFMT	EPMN_EPF	Espaces vides
EPFLG	EPAUCUN	0
EPPCFH	Noms et valeurs tels que définis dans <a href="#">Tableau 700, à la page 1125</a>	0

### **Remarque :**

1. Le symbole ↵ représente un caractère blanc unique.

## Déclaration RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D* MQEPH Structure
D*
D* Structure identifier
D EPSID          1          4
D* Structure version number
D EPVER          5          8I 0
D* Total length of MQEPH including MQCFHand parameter structures
D* that follow
D EPLEN          9          12I 0
D* Numeric encoding of data that follows last PCF parameter structure
D EPENC          13         16I 0
D* Character set identifier of data that follows last PCF parameter
D* structure
D EPCSI          17         20I 0
D* Format name of data that follows last PCF parameter structure
D EPFMT          21         28
D* Flags
D EPFLG          29         32I 0
D* Programmable Command Format Header
D EP3TYP          33         36I 0
D EP3LEN          37         40I 0
D EP3VER          41         44I 0
D EP3CMD          45         48I 0
D EP3SEQ          49         52I 0
D EP3CTL          53         56I 0
D EP3CC           57         60I 0
D EP3REA          61         64I 0
D EP3CNT          65         68I 0
```

IBM i

## MQGMO (options d'extraction de message) sous IBM i

La structure MQGMO permet à l'application de spécifier des options qui contrôlent la façon dont les messages sont supprimés des files d'attente.

### Présentation

#### Objet

La structure est un paramètre d'entrée-sortie dans l'appel MQGET.

#### Version

La version actuelle de MQGMO correspond à GMVER4. Les zones qui existent uniquement dans les versions les plus récentes de la structure sont identifiées comme telles dans les descriptions qui suivent.

Le fichier COPY fourni contient la version la plus récente de MQGMO qui est prise en charge par l'environnement, mais avec la valeur initiale de la zone *GMVER* définie sur *GMVER1*. Pour utiliser des zones qui ne sont pas présentes dans la structure version-1, l'application doit définir la zone *GMVER* sur le numéro de version de la version requise.

#### Jeu de caractères et codage

Les données dans MQGMO doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par ENNAT. Toutefois, si l'application s'exécute en tant que client IBM MQ, la structure doit être dans le jeu de caractères et le codage du client.

- «Zones», à la page [1127](#)
- «Valeurs initiales», à la page [1149](#)
- «Déclaration RPG», à la page [1149](#)

### Zones

La structure MQGMO contient les zones suivantes ; les zones sont décrites par ordre alphabétique:

### **GMGST (chaîne de caractères de 1 octet)**

Indicateur signalant si le message extrait se trouve dans un groupe.

Elle a l'une des valeurs suivantes :

#### **GSNIG**

Le message n'est pas dans un groupe.

#### **GSMIG**

Le message se trouve dans un groupe, mais n'est pas le dernier du groupe.

#### **GSLMIG**

Le message est le dernier du groupe.

Cette valeur est également la valeur renvoyée si le groupe se compose d'un seul message.

Cette zone est une zone de sortie. La valeur initiale de cette zone est GSNIG. Cette zone est ignorée si *GMVER* est inférieur à *GMVER2*.

### **GMMH (entier signé à 10 chiffres)**

descripteur de message

Si l'option *GMPRAQ* est spécifiée et que l'attribut de file d'attente *PRPCTL* n'est pas défini sur *PRPRFH*, il s'agit de l'identificateur d'un message qui est rempli avec les propriétés du message extrait de la file d'attente. Le descripteur est créé par un appel *MQCRTMH*. Toutes les propriétés déjà associées à l'identificateur sont effacées avant l'extraction d'un message.

La valeur suivante peut également être spécifiée:

*MQHM\_NONE*

Aucun descripteur de message fourni.

Aucun descripteur de message n'est requis dans l'appel *MQGET* si un descripteur de message valide est fourni et utilisé dans la sortie pour contenir les propriétés de message, le descripteur de message associé au descripteur de message est utilisé pour les zones d'entrée.

Si un descripteur de message est spécifié dans l'appel *MQGET*, il est toujours prioritaire sur le descripteur de message associé à un descripteur de message.

Si *GMPRRF* est indiqué ou si *GMPRAQ* est indiqué et que l'attribut de file d'attente *PRPCTL* est *PRPRFH*, l'appel échoue avec le code raison *RC2026* lorsqu'aucun paramètre de descripteur de message n'est indiqué.

Lors du retour de l'appel *MQGET*, les propriétés et le descripteur de message associés à cet identificateur de message sont mis à jour pour refléter l'état du message extrait (ainsi que le descripteur de message s'il a été fourni sur l'appel *MQGET*). Les propriétés du message peuvent ensuite être recherchées à l'aide de l'appel *MQINQMP*.

A l'exception des extensions de descripteur de message, lorsqu'elles sont présentes, une propriété qui peut être interrogée avec l'appel *MQINQMP* n'est pas contenue dans les données de message ; si le message dans la file d'attente contient des propriétés dans les données de message, elles sont supprimées des données de message avant que les données ne soient renvoyées à l'application.

Si aucun descripteur de message n'est fourni ou si la version est inférieure à *GMVER4*, vous devez fournir un descripteur de message valide dans l'appel *MQGET*. Toutes les propriétés de message (à l'exception de celles contenues dans le descripteur de message) sont renvoyées dans le sujet des données de message avec la valeur des options de propriété dans la structure *MQGMO* et l'attribut de file d'attente *PRPCTL*.

Cette zone est toujours une zone d'entrée. La valeur initiale de cette zone est *HMNONE*. Cette zone est ignorée si *GMVER* est inférieur à *GMVER4*.

### **GMMO (entier signé à 10 chiffres)**

Options contrôlant les critères de sélection utilisés pour *MQGET*.



Ces options permettent à l'application de choisir les zones du paramètre **MSGDSC** qui sont utilisées pour sélectionner le message renvoyé par l'appel MQGET. L'application définit les options requises dans cette zone, puis définit les zones correspondantes dans le paramètre **MSGDSC** sur les valeurs requises pour ces zones. Seuls les messages ayant ces valeurs dans le MQMD pour le message sont candidats à l'extraction à l'aide de ce paramètre **MSGDSC** dans l'appel MQGET. Les zones pour lesquelles l'option de correspondance correspondante n'est pas spécifiée sont ignorées lors de la sélection du message à renvoyer. Si aucun critère de sélection ne doit être utilisé dans l'appel MQGET (c'est-à-dire si un message est acceptable), *GMMO* doit être défini sur MONONE.

Si GMLOGO est spécifié, seuls certains messages peuvent être renvoyés par l'appel MQGET suivant:

- S'il n'y a pas de groupe ou de message logique en cours, seuls les messages dont la valeur est *MDSEQ* égale à 1 et *MDOFF* égale à 0 peuvent être renvoyés. Dans ce cas, une ou plusieurs des options suivantes peuvent être utilisées pour sélectionner lequel des messages éligibles est celui renvoyé:
  - MOMSGI
  - MOCORI
  - MOGRPI
- S'il existe un groupe ou un message logique en cours, seul le message suivant du groupe ou du segment suivant du message logique peut être renvoyé, ce qui ne peut pas être modifié par la spécification des options MO\*.

Dans les deux cas, les options de correspondance qui ne sont pas applicables peuvent toujours être spécifiées, mais la valeur de la zone appropriée dans le paramètre **MSGDSC** doit correspondre à la valeur de la zone correspondante dans le message à renvoyer ; l'appel échoue avec le code anomalie RC2247 si cette condition n'est pas satisfaite.

*GMMO* est ignoré si GMMUC ou GMBRWC est indiqué.

Une ou plusieurs des options suivantes peuvent être spécifiées:

#### **MOMSGI**

Extraire le message avec l'identificateur de message spécifié.

Cette option indique que le message à extraire doit avoir un identificateur de message qui correspond à la valeur de la zone *MDMID* dans le paramètre **MSGDSC** de l'appel MQGET. Cette correspondance s'ajoute à toute autre correspondance pouvant s'appliquer (par exemple, l'identificateur de corrélation).

Si cette option n'est pas spécifiée, la zone *MDMID* du paramètre **MSGDSC** est ignorée et tout identificateur de message correspond.

**Remarque :** L'identificateur de message MINONE est une valeur spéciale qui correspond à n'importe quel identificateur de message dans le MQMD du message. Par conséquent, la spécification de MOMSGI avec MINONE est identique à la non-spécification de MOMSGI.

#### **MOCORI**

Extraire le message avec l'identificateur de corrélation spécifié.

Cette option indique que le message à extraire doit avoir un identificateur de corrélation qui correspond à la valeur de la zone *MDCID* dans le paramètre **MSGDSC** de l'appel MQGET. Cette correspondance s'ajoute à toute autre correspondance pouvant s'appliquer (par exemple, l'identificateur de message).

Si cette option n'est pas spécifiée, la zone *MDCID* du paramètre **MSGDSC** est ignorée et tout identificateur de corrélation correspond.

**Remarque :** L'identificateur de corrélation CINONE est une valeur spéciale qui correspond à n'importe quel identificateur de corrélation dans le MQMD du message. Par conséquent, la spécification de MOCORI avec CINONE est identique à la non-spécification de MOCORI.

#### **MOGRPI**

Extraire le message avec l'identificateur de groupe spécifié.

Cette option indique que le message à extraire doit avoir un identificateur de groupe qui correspond à la valeur de la zone *MDGID* dans le paramètre **MSGDSC** de l'appel MQGET. Cette correspondance s'ajoute à toute autre correspondance pouvant s'appliquer (par exemple, l'identificateur de corrélation).

Si cette option n'est pas spécifiée, la zone *MDGID* du paramètre **MSGDSC** est ignorée et tout identificateur de groupe correspond.

**Remarque :** L'identificateur de groupe GINONE est une valeur spéciale qui correspond à tout identificateur de groupe dans le MQMD pour le message. Par conséquent, la spécification de MOGRPI avec GINONE est identique à la non-spécification de MOGRPI.

### **MOSEQN**

Extraire le message avec le numéro de séquence de message spécifié.

Cette option indique que le message à extraire doit avoir un numéro de séquence de message qui correspond à la valeur de la zone *MDSEQ* dans le paramètre **MSGDSC** de l'appel MQGET. Cette correspondance s'ajoute à toute autre correspondance pouvant s'appliquer (par exemple, l'identificateur de groupe).

Si cette option n'est pas spécifiée, la zone *MDSEQ* du paramètre **MSGDSC** est ignorée et tout numéro de séquence de message correspond.

### **MOOFFS**

Extraire le message avec le décalage spécifié.

Cette option indique que le message à extraire doit avoir un décalage qui correspond à la valeur de la zone *MDOFF* dans le paramètre **MSGDSC** de l'appel MQGET. Cette correspondance s'ajoute à toute autre correspondance pouvant s'appliquer (par exemple, le numéro de séquence du message).

Si cette option n'est pas spécifiée, la zone *MDOFF* du paramètre **MSGDSC** est ignorée et tout décalage correspond.

Si aucune des options décrites n'est spécifiée, l'option suivante peut être utilisée:

### **MOIS**

Aucune correspondance.

Cette option indique qu'aucune correspondance ne doit être utilisée lors de la sélection du message à renvoyer. Par conséquent, tous les messages de la file d'attente peuvent être extraits (mais soumis au contrôle des options GMAMSA, GMASGA et GMCMPM).

MONONE est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette option soit utilisée avec une autre option MO\*, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

Cette zone est une zone d'entrée. La valeur initiale de cette zone est MOMSGI avec MOCORI. Cette zone est ignorée si *GMVER* est inférieur à *GMVER2*.

**Remarque :** La valeur initiale de la zone *GMMO* est définie à des fins de compatibilité avec les gestionnaires de files d'attente de version antérieure. Toutefois, lors de la lecture d'une série de messages à partir d'une file d'attente sans utiliser de critères de sélection, cette valeur initiale nécessite que l'application réinitialise les zones *MDMID* et *MDCID* sur MINONE et CINONE avant chaque appel MQGET. La nécessité de réinitialiser *MDMID* et *MDCID* peut être évitée en définissant *GMVER* sur *GMVER2* et *GMMO* sur MONONE.

### **GMOPT (entier signé à 10 chiffres)**

Options qui contrôlent l'action de MQGET.

Aucune ou plusieurs des options décrites ci-après peuvent être spécifiées. Si plusieurs valeurs sont requises, les valeurs peuvent être ajoutées (n'ajoutez pas la même constante plus d'une fois). Les combinaisons d'options non valides sont notées ; toutes les autres combinaisons sont valides.

**Options d'attente:** Les options suivantes concernent l'attente de l'arrivée de messages dans la file d'attente:

### **Jeton Web GMWT**

Attendez que le message arrive.

L'application doit attendre l'arrivée d'un message approprié. La durée maximale d'attente de l'application est indiquée dans *GMWT*.

Si les demandes MQGET sont inhibées ou que les demandes MQGET sont inhibées pendant l'attente, l'attente est annulée et l'appel se termine avec CCFAIL et le code anomalie RC2016, qu'il y ait ou non des messages appropriés dans la file d'attente.

Cette option peut être utilisée avec les options GMBRWF ou GMBRWN.

Si plusieurs applications sont en attente dans la même file d'attente partagée, l'application ou les applications qui sont activées lorsqu'un message approprié arrive sont décrites plus loin dans cette section.

**Remarque :** Dans la description suivante, un appel MQGET de navigation est un appel qui spécifie l'une des options de navigation, mais pas GMLK ; un appel MQGET spécifiant l'option GMLK est traité comme un appel non navigant.

- Si un ou plusieurs appels MQGET non-navigation sont en attente, mais qu'aucun appel MQGET de navigation n'est en attente, un appel est activé.
- Si un ou plusieurs appels MQGET de navigation sont en attente, mais qu'aucun appel MQGET de non navigation n'est en attente, tous les appels sont activés.
- Si un ou plusieurs appels MQGET non navigants et un ou plusieurs appels MQGET de navigation sont en attente, un appel MQGET non navigant est activé et aucun, certains ou tous les appels MQGET de navigation sont activés. (Le nombre d'appels MQGET de navigation activés ne peut pas être prédit car il dépend des considérations de planification du système d'exploitation et d'autres facteurs.)

Si plusieurs appels MQGET non-browse sont en attente dans la même file d'attente, un seul est activé ; dans ce cas, le gestionnaire de files d'attente tente de donner la priorité aux appels non-browse en attente dans l'ordre suivant:

1. Demandes get-wait spécifiques qui ne peuvent être satisfaites que par certains messages, par exemple, ceux avec un *MDMID* ou un *MDCID* spécifique (ou les deux).
2. Demandes get-wait générales qui peuvent être satisfaites par n'importe quel message.

Il convient de noter les points suivants:

- Dans la première catégorie, aucune priorité supplémentaire n'est accordée aux demandes get-wait plus spécifiques, par exemple celles qui spécifient à la fois *MDMID* et *MDCID*.
- Dans l'une ou l'autre catégorie, il n'est pas possible de prédire quelle application est sélectionnée. En particulier, l'application en attente la plus longue n'est pas nécessairement celle sélectionnée.
- La longueur du chemin et les considérations relatives à la planification de la priorité du système d'exploitation peuvent signifier qu'une application en attente dont la priorité du système d'exploitation est inférieure à celle attendue extrait le message.
- Il peut également arriver qu'une application qui n'est pas en attente récupère le message de préférence à un autre.

Le GMWT est ignoré s'il est spécifié avec GMBRWC ou GMMUC ; aucune erreur n'est générée.

### **GMNWT**

Renvoyer immédiatement si aucun message ne convient.

L'application ne doit pas attendre si aucun message approprié n'est disponible. Il s'agit du contraire de l'option GMWT, qui est définie pour faciliter la documentation du programme. Il s'agit de la valeur par défaut si aucune n'est spécifiée.

## GMFIQ

Echec si le gestionnaire de files d'attente est mis au repos.

Cette option force l'échec de l'appel MQGET si le gestionnaire de files d'attente est à l'état de mise au repos.

Si cette option est spécifiée avec GMWT et que l'attente est en attente au moment où le gestionnaire de files d'attente passe à l'état de mise au repos:

- L'attente est annulée et l'appel renvoie le code achèvement CCFAIL avec le code anomalie RC2161 .

Si GMFIQ n'est pas spécifié et que le gestionnaire de files d'attente passe à l'état de mise au repos, l'attente n'est pas annulée.

**Options de point de synchronisation:** Les options suivantes concernent la participation de l'appel MQGET dans une unité d'oeuvre:

## GMSYP

Obtenir le message avec le contrôle de point de synchronisation.

La demande consiste à fonctionner dans le cadre des protocoles d'unité de travail normaux. Le message est marqué comme étant indisponible pour d'autres applications, mais il est supprimé de la file d'attente uniquement lorsque l'unité de travail est validée. Le message est de nouveau disponible si l'unité d'oeuvre est annulée.

Si cette option ou GMNSYP n'est pas indiquée, la demande d'obtention ne se trouve pas dans une unité d'oeuvre.

Cette option n'est pas valide avec l'une des options suivantes:

- MGMBRWF
- GMBRWC
- GMBRWN
- GMLK
- GMNSYP
- GMPSYP
- GMUNLK

## GMPSYP

Obtenir le message avec le contrôle de point de synchronisation si le message est persistant.

La demande doit fonctionner dans les protocoles d'unité de travail normaux, mais uniquement si le message extrait est persistant. Un message persistant a la valeur PEPER dans la zone *MDPER* de MQMD.

- Si le message est persistant, le gestionnaire de files d'attente traite l'appel comme si l'application avait spécifié GMSYP.
- Si le message n'est pas persistant, le gestionnaire de files d'attente traite l'appel comme si l'application avait spécifié GMNSYP (voir la section suivante pour plus de détails).

Cette option n'est pas valide avec l'une des options suivantes:

- MGMBRWF
- GMBRWC
- GMBRWN
- GMCMPM
- GMNSYP
- GMSYP
- GMUNLK

## **GMNSYP**

Obtenir un message sans contrôle de point de synchronisation.

La demande consiste à fonctionner en dehors des protocoles d'unité de travail normaux. Le message est immédiatement supprimé de la file d'attente (sauf s'il s'agit d'une demande de navigation). Le message ne peut pas être rendu disponible à nouveau en raison de l'annulation de l'unité d'oeuvre.

Cette option est prise en compte si GMBRWF ou GMBRWN est indiqué.

Si cette option et GMSYP ne sont pas spécifiées, la demande d'extraction ne se trouve pas dans une unité d'oeuvre.

Cette option n'est pas valide avec l'une des options suivantes:

- GMSYP
- GMPSYP

**Options de navigation:** Les options suivantes concernent la navigation dans les messages de la file d'attente:

## **MGMBRWF**

Parcourez depuis le début de la file d'attente.

Lorsqu'une file d'attente est ouverte avec l'option OOBROW, un curseur de navigation est établi, positionné logiquement avant le premier message de la file d'attente. Les appels MQGET ultérieurs spécifiant l'option GMBRWF, GMBRWN ou GMBRWC peuvent être utilisés pour extraire des messages de la file d'attente de façon non destructive. Le curseur de navigation marque la position, dans les messages de la file d'attente, à partir de laquelle le prochain appel MQGET avec GMBRWN recherche un message approprié.

Un appel MQGET avec GMBRWF entraîne la non prise en compte de la position précédente du curseur de navigation. Le premier message de la file d'attente qui remplit les conditions spécifiées dans le descripteur de message est extrait. Le message reste dans la file d'attente et le curseur de navigation est positionné sur ce message.

Après cet appel, le curseur de navigation est positionné sur le message qui a été renvoyé. Si le message est supprimé de la file d'attente avant l'émission du prochain appel MQGET avec GMBRWN, le curseur de navigation reste à l'emplacement de la file d'attente occupé par le message, même si cette position est à présent vide.

L'option GMMUC peut ensuite être utilisée avec un appel MQGET non-navigant, si nécessaire, pour supprimer le message de la file d'attente.

Le curseur de navigation n'est pas déplacé par un appel MQGET non navigant utilisant le même descripteur *HOBJ*. Il n'est pas non plus déplacé par un appel MQGET de navigation qui renvoie le code achèvement CCFAIL ou le code anomalie RC2080.

L'option GMLK peut être spécifiée avec cette option pour que le message consulté soit verrouillé.

GMBRWF peut être indiqué avec n'importe quelle combinaison correcte des options GM\* et MO\* qui contrôlent le traitement des messages dans les groupes et les segments de messages logiques.

Si GMLOGO est spécifié, les messages sont consultés dans l'ordre logique. Si cette option est omise, les messages sont consultés dans l'ordre physique. Lorsque GMBRWF est spécifié, il est possible de passer de l'ordre logique à l'ordre physique, mais les appels MQGET ultérieurs utilisant GMBRWN doivent parcourir la file d'attente dans le même ordre que l'appel le plus récent ayant spécifié GMBRWF pour l'identificateur de file d'attente.

Les informations de groupe et de segment conservées par le gestionnaire de files d'attente pour les appels MQGET qui parcourent les messages de la file d'attente sont distinctes des informations de groupe et de segment conservées par le gestionnaire de files d'attente pour les appels MQGET qui suppriment les messages de la file d'attente. Lorsque GMBRWF est spécifié, le gestionnaire de files d'attente ignore les informations de groupe et de segment pour

la consultation et analyse la file d'attente comme s'il n'y avait pas de groupe en cours et de message logique en cours. Si l'appel MQGET aboutit (code achèvement CCOK ou CCWARN), les informations de groupe et de segment pour la navigation sont définies sur celles du message renvoyé ; si l'appel échoue, les informations de groupe et de segment restent les mêmes qu'avant l'appel.

Cette option n'est pas valide avec l'une des options suivantes:

- GMBRWC
- GMBRWN
- UC de la GMMUC
- GMSYP
- GMPSYP
- GMUNLK

Il s'agit également d'une erreur si la file d'attente n'a pas été ouverte pour consultation.

### **GMBRWN**

Parcourir à partir de la position en cours dans la file d'attente.

Le curseur de navigation est avancé jusqu'au message suivant de la file d'attente qui répond aux critères de sélection spécifiés dans l'appel MQGET. Le message est renvoyé à l'application, mais reste dans la file d'attente.

Une fois qu'une file d'attente a été ouverte pour consultation, le premier appel de consultation à l'aide de l'identificateur a le même effet, qu'il indique l'option GMBRWF ou GMBRWN.

Si le message est supprimé de la file d'attente avant l'émission du prochain appel MQGET avec GMBRWN, le curseur de navigation reste logiquement à l'emplacement de la file d'attente occupé par le message, même si cette position est vide.

Les messages sont stockés dans la file d'attente de deux manières:

- FIFO dans la priorité (MSPRIO), ou
- FIFO quelle que soit la priorité (MSFIFO)

L'attribut de file d'attente **MsgDeliverySequence** indique la méthode qui s'applique (voir [«Attributs des files d'attente»](#), à la page 1435 pour plus de détails).

Si la file d'attente a une valeur *MsgDeliverySequence* de MSPRIO et qu'un message arrive dans la file d'attente dont la priorité est supérieure à celle indiquée par le curseur de navigation, ce message est introuvable lors du balayage en cours de la file d'attente via GMBRWN. Il ne peut être trouvé qu'une fois que le curseur de navigation a été réinitialisé avec GMBRWF (ou en rouvrant la file d'attente).

L'option GMMUC peut être utilisée ultérieurement avec un appel MQGET de non-consultation, si nécessaire, pour supprimer le message de la file d'attente.

Le curseur de navigation n'est pas déplacé par des appels MQGET non navigants utilisant le même descripteur *HOB*J .

L'option GMLK peut être spécifiée avec cette option pour que le message consulté soit verrouillé.

Vous pouvez indiquer GMBRWN avec n'importe quelle combinaison correcte des options GM\* et MO\* qui contrôlent le traitement des messages dans les groupes et les segments de messages logiques.

Si GMLOGO est spécifié, les messages sont consultés dans l'ordre logique. Si cette option est omise, les messages sont consultés dans l'ordre physique. Lorsque GMBRWF est spécifié, il est possible de passer de l'ordre logique à l'ordre physique, mais les appels MQGET ultérieurs utilisant GMBRWN doivent parcourir la file d'attente dans le même ordre que l'appel le plus récent ayant spécifié GMBRWF pour l'identificateur de file d'attente. L'appel échoue avec le code anomalie RC2259 si cette condition n'est pas satisfaite.

**Remarque :** Une attention particulière est nécessaire si un appel MQGET est utilisé pour parcourir au-delà de la fin d'un groupe de messages (ou d'un message logique ne figurant pas dans un groupe) lorsque GMLOGO n'est pas spécifié. Par exemple, si le dernier message du groupe précède le premier message du groupe dans la file d'attente, en utilisant GMBRWN pour parcourir la fin du groupe, en spécifiant MOSEQN avec MDSEQ défini sur 1 (pour trouver le premier message du groupe suivant), le premier message du groupe déjà consulté sera renvoyé. Cela peut se produire immédiatement ou un certain nombre d'appels MQGET ultérieurement (s'il existe des groupes intermédiaires).

La possibilité d'une boucle sans fin peut être évitée en ouvrant la file d'attente deux fois pour parcourir:

- Utilisez le premier descripteur pour parcourir uniquement le premier message de chaque groupe.
- Utilisez le deuxième descripteur pour parcourir uniquement les messages d'un groupe spécifique.
- Utilisez les options MO\* pour déplacer le deuxième curseur de navigation à la position du premier curseur de navigation, avant de parcourir les messages du groupe.
- N'utilisez pas GMBRWN pour naviguer au-delà de la fin d'un groupe.

Les informations de groupe et de segment que le gestionnaire de files d'attente conserve pour les appels MQGET qui parcourent les messages de la file d'attente sont distinctes des informations de groupe et de segment qu'il conserve pour les appels MQGET qui suppriment les messages de la file d'attente.

Cette option n'est pas valide avec l'une des options suivantes:

- MGMBRWF
- GMBRWC
- UC de la GMMUC
- GMSYP
- GMPSYP
- GMUNLK

Il s'agit également d'une erreur si la file d'attente n'a pas été ouverte pour consultation.

## **GMBRWC**

Parcourir le message sous le curseur de navigation.

Cette option entraîne l'extraction non destructive du message pointé par le curseur de navigation, quelles que soient les options MO\* spécifiées dans la zone GMMO de MQGMO.

Le message désigné par le curseur de navigation est celui qui a été rappelé pour la dernière fois à l'aide de l'option GMBRWF ou GMBRWN. L'appel échoue si aucun de ces appels n'a été émis pour cette file d'attente depuis son ouverture ou si le message qui se trouvait sous le curseur de navigation a été extrait de façon destructive.

La position du curseur de navigation n'est pas modifiée par cet appel.

L'option GMMUC peut ensuite être utilisée avec un appel MQGET non-navigant, si nécessaire, pour supprimer le message de la file d'attente.

Le curseur de navigation n'est pas déplacé par un appel MQGET non navigant utilisant le même descripteur *HOB*. Il n'est pas non plus déplacé par un appel MQGET de recherche qui renvoie le code achèvement CCFAIL ou le code anomalie RC2080.

Si GMBRWC est spécifié avec GMLK:

- Si un message est déjà verrouillé, il doit être celui qui se trouve sous le curseur, de sorte qu'il est renvoyé sans déverrouillage ni reverrouillage ; le message reste verrouillé.

- S'il n'y a pas de message verrouillé, le message sous le curseur de navigation (s'il y en a un) est verrouillé et renvoyé à l'application ; s'il n'y a pas de message sous le curseur de navigation, l'appel échoue.

Si GMBRWC est spécifié sans GMLK:

- Si un message est déjà verrouillé, il doit être celui qui se trouve sous le curseur. Ce message est renvoyé à l'application, puis déverrouillé. Etant donné que le message est maintenant déverrouillé, il n'est pas garanti qu'il puisse être consulté à nouveau ou extrait de façon destructive (il peut être extrait de façon destructive par une autre application obtenant des messages de la file d'attente).
- S'il n'y a pas de message verrouillé, le message sous le curseur de navigation (s'il y en a un) est renvoyé à l'application ; s'il n'y a pas de message sous le curseur de navigation, l'appel échoue.

Si GMCMPM est spécifié avec GMBRWC, le curseur de navigation doit identifier un message avec une zone *MDOFF* dans MQMD égale à zéro. Si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie RC2246 .

Les informations de groupe et de segment que le gestionnaire de files d'attente conserve pour les appels MQGET qui parcourent les messages de la file d'attente sont distinctes des informations de groupe et de segment qu'il conserve pour les appels MQGET qui suppriment les messages de la file d'attente.

Cette option n'est pas valide avec l'une des options suivantes:

- MGMBRWF
- GMBRWN
- UC de la GMMUC
- GMSYP
- GMPSYP
- GMUNLK

Il s'agit également d'une erreur si la file d'attente n'a pas été ouverte pour consultation.

### **UC de la GMMUC**

Obtenir le message sous le curseur de navigation.

Cette option permet d'extraire le message pointé par le curseur de navigation, quelles que soient les options MO\* spécifiées dans la zone *GMMO* de MQGMO. Le message est supprimé de la file d'attente.

Le message désigné par le curseur de navigation est celui qui a été rappelé pour la dernière fois à l'aide de l'option GMBRWF ou GMBRWN.

Si GMCMPM est spécifié avec GMMUC, le curseur de navigation doit identifier un message avec une zone *MDOFF* dans MQMD égale à zéro. Si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie RC2246 .

Cette option n'est pas valide avec l'une des options suivantes:

- MGMBRWF
- GMBRWC
- GMBRWN
- GMUNLK

Il s'agit également d'une erreur si la file d'attente n'a pas été ouverte à la fois pour l'exploration et pour l'entrée. Si le curseur de navigation ne pointe pas vers un message extractible, une erreur est renvoyée par l'appel MQGET.

**Options de verrouillage:** Les options suivantes concernent le verrouillage des messages dans la file d'attente:



## GMLK

Verrouiller le message.

Cette option verrouille le message consulté, de sorte que le message devient invisible pour tout autre descripteur ouvert pour la file d'attente. L'option ne peut être spécifiée que si l'une des options suivantes est également spécifiée:

- MGMBRWF
- GMBRWN
- GMBRWC

Un seul message peut être verrouillé par descripteur de file d'attente, mais il peut s'agir d'un message logique ou d'un message physique:

- Si GMCMPM est spécifié, tous les segments de message qui constituent le message logique sont verrouillés sur le descripteur de file d'attente (s'ils sont tous présents dans la file d'attente et disponibles pour l'extraction).
- Si GMCMPM n'est pas indiqué, un seul message physique est verrouillé pour l'identificateur de file d'attente. Si ce message est un segment d'un message logique, le segment verrouillé empêche d'autres applications d'utiliser GMCMPM pour extraire ou parcourir le message logique.

Le message verrouillé est toujours celui qui se trouve sous le curseur de navigation, et le message peut être supprimé de la file d'attente par un appel MQGET ultérieur qui spécifie l'option GMMUC. D'autres appels MQGET utilisant le descripteur de file d'attente peuvent également supprimer le message (par exemple, un appel qui spécifie l'identificateur du message verrouillé).

Si l'appel renvoie le code achèvement CCFAIL ou CCWARN avec le code anomalie RC2080, aucun message n'est verrouillé.

Si l'application décide de ne pas supprimer le message de la file d'attente, le verrou est libéré par:

- Emission d'un autre appel MQGET pour ce descripteur, avec GMBRWF ou GMBRWN spécifiés (avec ou sans GMLK) ; le message est déverrouillé si l'appel se termine avec CCOK ou CCWARN, mais reste verrouillé si l'appel se termine avec CCFAIL. Cependant, les exceptions suivantes s'appliquent :
  - Le message n'est pas déverrouillé si CCWARN est renvoyé avec RC2080.
  - Le message est déverrouillé si CCFAIL est renvoyé avec RC2033.

Si GMLK est également spécifié, le message renvoyé est verrouillé. Si GMLK n'est pas spécifié, il n'y a pas de message verrouillé après l'appel.

Si GMWT est spécifié et qu'aucun message n'est immédiatement disponible, le déverrouillage du message d'origine se produit avant le début de l'attente (à condition que l'appel soit exempt d'erreur).

- Emission d'un autre appel MQGET pour ce descripteur, avec GMBRWC (sans GMLK) ; le message est déverrouillé si l'appel se termine avec CCOK ou CCWARN, mais reste verrouillé si l'appel se termine avec CCFAIL. Toutefois, l'exception suivante s'applique:
  - Le message n'est pas déverrouillé si CCWARN est renvoyé avec RC2080.
- Emission d'un autre appel MQGET pour cet identificateur avec GMUNLK.
- Emission d'un appel MQCLOSE pour ce descripteur (explicitement ou implicitement par l'arrêt de l'application).

Aucune option d'ouverture spéciale n'est requise pour indiquer cette option, autre que OOBW, qui est nécessaire pour indiquer l'option de navigation associée.

Cette option n'est pas valide avec l'une des options suivantes:

- GMSYP
- GMPSYP

- GMUNLK

### **GMUNLK**

Déverrouiller le message.

Le message à déverrouiller doit avoir été préalablement verrouillé par un appel MQGET avec l'option GMLK. Si aucun message n'est verrouillé pour cet identificateur, l'appel se termine avec CCWARN et RC2209 .

Les paramètres **MSGDSC**, **BUFLEN**, **BUFFER** et **DATLEN** ne sont pas vérifiés ou modifiés si GMUNLK est spécifié. Aucun message n'est renvoyé dans *BUFFER*.

Aucune option d'ouverture spéciale n'est requise pour spécifier cette option (bien que OOBW soit nécessaire pour émettre la demande de verrouillage en premier lieu).

Cette option n'est valide qu'avec les options suivantes:

- GMNWT
- GMNSYP

Ces deux options sont supposées être spécifiées ou non.

**Options de données de message:** Les options suivantes concernent le traitement des données de message lorsque le message est lu à partir de la file d'attente:

### **GMATM**

Autoriser la troncature des données de message.

Si la mémoire tampon de messages est trop petite pour contenir le message complet, cette option permet à l'appel MQGET de remplir la mémoire tampon avec autant de messages que la mémoire tampon peut contenir, d'émettre un code achèvement d'avertissement et de terminer son traitement. Cela signifie:

- Lors de la navigation dans les messages, le curseur de navigation est avancé jusqu'au message renvoyé.
- Lors de la suppression de messages, le message renvoyé est supprimé de la file d'attente.
- Le code anomalie RC2079 est renvoyé si aucune autre erreur ne se produit.

Sans cette option, la mémoire tampon est toujours remplie avec autant de messages qu'elle peut contenir, un code d'achèvement d'avertissement est émis, mais le traitement n'est pas terminé. Cela signifie:

- Lors de la navigation dans les messages, le curseur de navigation n'est pas avancé.
- Lors de la suppression de messages, le message n'est pas supprimé de la file d'attente.
- Le code anomalie RC2080 est renvoyé si aucune autre erreur ne se produit.

### **CONV MB**

Convertissez les données de message.

Cette option demande que les données d'application du message soient converties, afin de se conformer aux valeurs *MDCSI* et *MDENC* spécifiées dans le paramètre **MSGDSC** de l'appel MQGET, avant que les données ne soient copiées dans le paramètre **BUFFER** .

La zone *MDFMT* spécifiée lors de l'insertion du message est prise en compte par le processus de conversion pour identifier la nature des données du message. La conversion des données de message est effectuée par le gestionnaire de files d'attente pour les formats intégrés et par un exit écrit par l'utilisateur pour les autres formats.

- Si la conversion aboutit, les zones *MDCSI* et *MDENC* spécifiées dans le paramètre **MSGDSC** sont inchangées lors du retour de l'appel MQGET.
- Si la conversion ne peut pas être effectuée correctement (mais que l'appel MQGET se termine sans erreur), les données de message sont renvoyées non converties et les zones *MDCSI* et *MDENC* dans *MSGDSC* sont définies sur les valeurs du message non converti. Le code achèvement est CCWARN dans ce cas.

Dans les deux cas, ces zones décrivent l'identificateur de jeu de caractères et le codage des données de message renvoyées dans le paramètre **BUFFER**.

Voir la zone *MDFMT* décrite dans «MQMD (descripteur de message) sous IBM i», à la page 1162 pour la liste des noms de format pour lesquels le gestionnaire de files d'attente effectue la conversion.

**Options de groupe et de segment:** Les options suivantes concernent le traitement des messages dans des groupes et des segments de messages logiques. Ces définitions peuvent vous aider à comprendre les options suivantes:

### **Message physique**

Il s'agit de la plus petite unité d'informations pouvant être placée dans ou supprimée d'une file d'attente ; elle correspond souvent aux informations spécifiées ou extraites sur un seul appel MQPUT, MQPUT1 ou MQGET. Chaque message physique possède son propre descripteur de message (MQMD). En règle générale, les messages physiques se distinguent par des valeurs différentes pour l'identificateur de message (*zoneMDMID* dans MQMD), bien que cela ne soit pas appliqué par le gestionnaire de files d'attente.

### **Message logique**

Il s'agit d'une unité unique d'informations d'application. En l'absence de contraintes système, un message logique serait identique à un message physique. Mais lorsque les messages logiques sont volumineux, les contraintes du système peuvent rendre souhaitable ou nécessaire la division d'un message logique en deux ou plusieurs messages physiques, appelés segments.

Un message logique segmenté se compose d'au moins deux messages physiques ayant le même identificateur de groupe non null (*zoneMDGID* dans MQMD) et le même numéro de séquence de message (*zoneMDSEQ* dans MQMD). Les segments se distinguent par des valeurs différentes pour le décalage de segment (*zoneMDOFF* dans MQMD), qui fournit le décalage des données dans le message physique à partir du début des données dans le message logique. Chaque segment étant un message physique, les segments d'un message logique ont généralement des identificateurs de message différents.

Un message logique qui n'a pas été segmenté, mais pour lequel la segmentation a été autorisée par l'application émettrice, possède également un identificateur de groupe non null, bien que dans ce cas, il n'y ait qu'un seul message physique avec cet identificateur de groupe si le message logique n'appartient pas à un groupe de messages. Les messages logiques pour lesquels la segmentation a été inhibée par l'application émettrice ont un identificateur de groupe nul (GINONE), sauf si le message logique appartient à un groupe de messages.

### **Groupe de messages**

Il s'agit d'un ensemble d'un ou de plusieurs messages logiques ayant le même identificateur de groupe non null. Les messages logiques du groupe se distinguent par des valeurs différentes pour le numéro de séquence de message, qui est un entier compris entre 1 et n, où n est le nombre de messages logiques du groupe. Si un ou plusieurs des messages logiques sont segmentés, il y a plus de n messages physiques dans le groupe.

### **MGMLOGO**

Les messages des groupes et des segments de messages logiques sont renvoyés dans l'ordre logique.

Cette option contrôle l'ordre dans lequel les messages sont renvoyés par les appels MQGET successifs pour le descripteur de file d'attente. L'option doit être spécifiée sur chacun de ces appels afin d'avoir un effet.

Si GMLOGO est spécifié pour les appels MQGET successifs de l'identificateur de file d'attente, les messages des groupes sont renvoyés dans l'ordre indiqué par leurs numéros de séquence de messages et les segments de messages logiques sont renvoyés dans l'ordre indiqué par leurs décalages de segment. Cet ordre peut être différent de l'ordre dans lequel ces messages et segments apparaissent dans la file d'attente.

**Remarque :** La spécification de GMLOGO n'a aucune conséquence négative sur les messages qui n'appartiennent pas à des groupes et qui ne sont pas des segments. En effet, ces messages sont

traités comme si chacun appartenait à un groupe de messages composé d'un seul message. Par conséquent, il est parfaitement sûr de spécifier GMLOGO lors de l'extraction de messages à partir de files d'attente qui peuvent contenir un mélange de messages dans des groupes, des segments de message et des messages non segmentés dans des groupes.

Pour renvoyer les messages dans l'ordre requis, le gestionnaire de files d'attente conserve les informations de groupe et de segment entre les appels MQGET successifs. Ces informations identifient le groupe de messages en cours et le message logique en cours pour l'identificateur de file d'attente, la position en cours dans le groupe et le message logique, et indiquent si les messages sont extraits dans une unité d'oeuvre. Etant donné que le gestionnaire de files d'attente conserve ces informations, l'application n'a pas besoin de définir les informations de groupe et de segment avant chaque appel MQGET. En particulier, cela signifie que l'application n'a pas besoin de définir les zones *MDGID*, *MDSEQ* et *MDOFF* dans MQMD. Toutefois, l'application doit définir correctement l'option GMSYP ou GMNSYP à chaque appel.

Lorsque la file d'attente est ouverte, il n'y a pas de groupe de messages en cours ni de message logique en cours. Un groupe de messages devient le groupe de messages en cours lorsqu'un message comportant l'indicateur MFMIG est renvoyé par l'appel MQGET. Lorsque GMLOGO est spécifié lors d'appels successifs, ce groupe reste le groupe en cours jusqu'à ce qu'un message contenant:

- MFLMIG sans MFSEG (c'est-à-dire que le dernier message logique du groupe n'est pas segmenté), ou
- MFLMIG avec MFLSEG (c'est-à-dire que le message renvoyé est le dernier segment du dernier message logique du groupe).

Lorsqu'un tel message est renvoyé, le groupe de messages est arrêté et, à la fin de cet appel MQGET, il n'y a plus de groupe en cours. De la même manière, un message logique devient le message logique en cours lorsqu'un message comportant l'indicateur MFSEG est renvoyé par l'appel MQGET et que ce message logique est arrêté lorsque le message comportant l'indicateur MFLSEG est renvoyé.

Si aucun critère de sélection n'est spécifié, les appels MQGET successifs renvoient (dans l'ordre correct) les messages du premier groupe de messages de la file d'attente, puis les messages du deuxième groupe de messages, etc., jusqu'à ce qu'il n'y ait plus de messages disponibles. Il est possible de sélectionner les groupes de messages particuliers renvoyés en spécifiant une ou plusieurs des options suivantes dans la zone *GMMO* :

- MOMSGI
- MOCORI
- MOGRPI

Toutefois, ces options ne sont effectives que lorsqu'il n'y a pas de groupe de messages en cours ou de message logique ; voir la zone *GMMO* décrite dans cette rubrique.

Tableau 702, à la page 1141 affiche les valeurs des zones *MDMID*, *MDCID*, *MDGID*, *MDSEQ* et *MDOFF* que le gestionnaire de files d'attente recherche lorsqu'il tente de trouver un message à renvoyer lors de l'appel MQGET. Cela s'applique à la fois à la suppression de messages de la file d'attente et à la recherche de messages dans la file d'attente. Les colonnes de la table ont les significations suivantes:

**NOM DE JOURNAL**

Indique si l'option GMLOGO est spécifiée lors de l'appel.

**Groupe de travail en cours**

Indique si un groupe de messages en cours existe avant l'appel.

**Message du journal en cours**

Indique si un message logique en cours existe avant l'appel.

**Autres colonnes**

Affiche les valeurs que le gestionnaire de files d'attente recherche. "Précédent" indique la valeur renvoyée pour la zone dans le message précédent pour l'identificateur de file d'attente.

Tableau 702. Options MQGET relatives aux messages dans les groupes et les segments de messages logiques

Options que vous spécifiez	Statut du groupe et du message de journal avant l'appel		Valeurs que le gestionnaire de files d'attente recherche				
	Groupe en cours	Message du journal en cours	MDMID	MDCID	MDGID	MDSEQ	MDOFF
Oui	Non	Non	Contrôlé par GMMO	Contrôlé par GMMO	Contrôlé par GMMO	1	0
Oui	Non	Oui	Tout identificateur de message	Tout identificateur de corrélation	Identificateur de groupe précédent	1	Décalage précédent + longueur de segment précédent
Oui	Oui	Non	Tout identificateur de message	Tout identificateur de corrélation	Identificateur de groupe précédent	Numéro de séquence précédent + 1	0
Oui	Oui	Oui	Tout identificateur de message	Tout identificateur de corrélation	Identificateur de groupe précédent	Numéro de séquence précédent	Décalage précédent + longueur de segment précédent
Non	L'un ou l'autre	L'un ou l'autre	Contrôlé par GMMO	Contrôlé par GMMO	Contrôlé par GMMO	Contrôlé par GMMO	Contrôlé par GMMO

Lorsque plusieurs groupes de messages sont présents dans la file d'attente et peuvent être renvoyés, ils sont renvoyés dans l'ordre déterminé par la position dans la file d'attente du premier segment du premier message logique de chaque groupe (c'est-à-dire, les messages physiques dont les numéros de séquence de message sont 1 et les décalages de 0, déterminent l'ordre dans lequel les groupes éligibles sont renvoyés).

L'option GMLOGO affecte les unités de travail comme suit:

- Si le premier message logique ou segment d'un groupe est extrait dans une unité d'oeuvre, tous les autres messages logiques et segments du groupe doivent être extraits dans une unité d'oeuvre, si le même descripteur de file d'attente est utilisé. Toutefois, il n'est pas nécessaire de les extraire dans la même unité de travail. Cela permet à un groupe de messages composé de plusieurs messages physiques d'être scindé en plusieurs unités de travail consécutives pour l'identificateur de file d'attente.
- Si le premier message logique ou segment d'un groupe n'est pas extrait dans une unité d'oeuvre, aucun des autres messages logiques et segments du groupe ne peut être extrait dans une unité d'oeuvre, si le même descripteur de file d'attente est utilisé.

Si ces conditions ne sont pas satisfaites, l'appel MQGET échoue avec le code anomalie RC2245 .

Lorsque GMLOGO est spécifié, le MQGMO fourni dans l'appel MQGET ne doit pas être inférieur à GMVER2 et le MQMD ne doit pas être inférieur à MDVER2. Si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie RC2256 ou RC2257 , selon le cas.

Si GMLOGO n'est pas spécifié pour les appels MQGET successifs du descripteur de file d'attente, les messages sont renvoyés indépendamment de leur appartenance à des groupes de messages

ou de leur appartenance à des segments de messages logiques. Cela signifie que les messages ou les segments d'un groupe ou d'un message logique particulier peuvent être renvoyés dans l'ordre, ou être mélangés avec des messages ou des segments d'autres groupes ou messages logiques, ou avec des messages qui ne sont pas dans des groupes et qui ne sont pas des segments. Dans cette situation, les messages particuliers renvoyés par les appels MQGET successifs sont contrôlés par les options MO\* spécifiées sur ces appels (voir la zone *GMMO* décrite dans «MQGMO (options d'extraction de message) sous IBM i», à la page 1127 pour plus de détails sur ces options).

Il s'agit de la technique qui peut être utilisée pour redémarrer un groupe de messages ou un message logique au milieu, après une défaillance du système. Lorsque le système redémarre, l'application peut définir les zones *MDGID*, *MDSEQ*, *MDOFF* et *GMMO* sur les valeurs appropriées, puis émettre l'appel MQGET avec *GMSYP* ou *GMNSYP* défini selon les besoins, mais sans spécifier *GMLOGO*. Si cet appel aboutit, le gestionnaire de files d'attente conserve les informations de groupe et de segment, et les appels MQGET suivants utilisant ce descripteur de file d'attente peuvent spécifier *GMLOGO* comme normal.

Les informations de groupe et de segment que le gestionnaire de files d'attente conserve pour l'appel MQGET sont distinctes des informations de groupe et de segment qu'il conserve pour l'appel MQPUT. En outre, le gestionnaire de files d'attente conserve des informations distinctes pour:

- Appels MQGET qui suppriment des messages de la file d'attente.
- Appels MQGET qui parcourent les messages de la file d'attente.

Pour un descripteur de file d'attente donné, l'application est libre de mélanger les appels MQGET qui spécifient *GMLOGO* avec les appels MQGET qui ne le font pas, mais les points suivants doivent être notés:

- Si *GMLOGO* n'est pas spécifié, chaque appel MQGET réussi entraîne le gestionnaire de files d'attente à définir les informations de groupe et de segment sauvegardées sur les valeurs correspondant au message renvoyé ; cela remplace les informations de groupe et de segment existantes conservées par le gestionnaire de files d'attente pour l'identificateur de file d'attente. Seules les informations appropriées à l'action de l'appel (parcourir ou supprimer) sont modifiées.
- Si *GMLOGO* n'est pas spécifié, l'appel n'échoue pas s'il existe un groupe de messages en cours ou un message logique ; l'appel peut toutefois aboutir avec un code achèvement *CCWARN*. Le Tableau 703, à la page 1142 montre les différentes observations qui peuvent survenir. Dans ces cas, si le code achèvement n'est pas *CCOK*, le code anomalie est l'un des suivants:
  - RC2241
  - RC2242
  - RC2245

**Remarque :** Le gestionnaire de files d'attente ne vérifie pas les informations de groupe et de segment lors de la consultation d'une file d'attente ou lors de la fermeture d'une file d'attente qui a été ouverte pour consultation mais pas d'entrée ; dans ces cas, le code achèvement est toujours *CCOK* (en supposant qu'il n'y a pas d'autres erreurs).

<i>Tableau 703. Résultat lorsque l'appel MQGET ou MQCLOSE n'est pas cohérent avec les informations de groupe et de segment</i>		
<b>L'appel en cours est</b>	<b>L'appel précédent était MQGET avec GMLOGO</b>	<b>L'appel précédent était MQGET sans GMLOGO</b>
MQGET avec GMLOGO	CCFAIL	CCFAIL
MQGET sans GMLOGO	CCWARN	CCOK

Tableau 703. Résultat lorsque l'appel MQGET ou MQCLOSE n'est pas cohérent avec les informations de groupe et de segment (suite)

L'appel en cours est	L'appel précédent était MQGET avec GMLOGO	L'appel précédent était MQGET sans GMLOGO
MQCLOSE avec un groupe non terminé ou un message logique	CCWARN	CCOK

Les applications qui souhaitent simplement extraire des messages et des segments dans l'ordre logique sont recommandées pour spécifier GMLOGO, car il s'agit de l'option la plus simple à utiliser. Cette option dispense l'application de la nécessité de gérer les informations de groupe et de segment, car le gestionnaire de files d'attente gère ces informations. Cependant, les applications spécialisées peuvent avoir besoin d'un contrôle plus important que celui fourni par l'option GMLOGO, et cela peut être réalisé en ne spécifiant pas cette option. Dans ce cas, l'application doit s'assurer que les zones *MDMID*, *MDCID*, *MDGID*, *MDSEQ* et *MDOFF* dans MQMD, ainsi que les options MO\* dans GMMO dans MQGMO, sont correctement définies, avant chaque appel MQGET.

Par exemple, une application qui souhaite transmettre des messages physiques qu'elle reçoit, sans tenir compte du fait que ces messages se trouvent dans des groupes ou des segments de messages logiques, ne doit pas spécifier GMLOGO. En effet, dans un réseau complexe avec plusieurs chemins entre les gestionnaires de files d'attente d'envoi et de réception, les messages physiques peuvent arriver dans le désordre. En ne spécifiant pas GMLOGO et le PMLOGO correspondant sur l'appel MQPUT, l'application de réacheminement peut extraire et réacheminer chaque message physique dès qu'il arrive, sans avoir à attendre le message suivant dans l'ordre logique d'arrivée.

GMLOGO peut être spécifié avec n'importe quelle autre option GM\*, et avec diverses options MO\* dans des circonstances appropriées.

### GMCMPPM

Seuls les messages logiques complets peuvent être extraits.

Cette option indique que seul un message logique complet peut être renvoyé par l'appel MQGET. Si le message logique est segmenté, le gestionnaire de files d'attente réassemble les segments et renvoie le message logique complet à l'application. Le fait que le message logique ait été segmenté n'est pas évident pour l'application qui l'extrait.

**Remarque :** Il s'agit de la seule option qui permet au gestionnaire de files d'attente de réassembler des segments de message. S'ils ne sont pas spécifiés, les segments sont renvoyés individuellement à l'application s'ils sont présents dans la file d'attente (et répondent aux autres critères de sélection spécifiés dans l'appel MQGET). Les applications qui ne souhaitent pas recevoir de segments individuels doivent donc toujours spécifier GMCMPPM.

Pour utiliser cette option, l'application doit fournir une mémoire tampon suffisamment grande pour recevoir le message complet, ou spécifier l'option GMATM.

Si la file d'attente contient des messages segmentés dont certains des segments sont manquants (peut-être parce qu'ils ont été retardés dans le réseau et qu'ils ne sont pas encore arrivés), la spécification de GMCMPPM empêche l'extraction de segments appartenant à des messages logiques incomplets. Toutefois, ces segments de message contribuent toujours à la valeur de l'attribut de file d'attente **CurrentQDepth** ; cela signifie qu'il peut n'y avoir aucun message logique récupérable, même si *CurrentQDepth* est supérieur à zéro.

Pour les messages persistants, le gestionnaire de files d'attente peut réassembler les segments uniquement dans une unité de travail:

- Si l'appel MQGET fonctionne dans une unité de travail définie par l'utilisateur, cette unité de travail est utilisée. Si l'appel échoue tout au long du processus de réassemblage, le gestionnaire

de files d'attente rétablit dans la file d'attente tous les segments qui ont été supprimés lors du réassemblage. Toutefois, l'échec n'empêche pas la validation de l'unité de travail.

- Si l'appel est effectué en dehors d'une unité de travail définie par l'utilisateur et qu'il n'existe aucune unité de travail définie par l'utilisateur, le gestionnaire de files d'attente crée une unité de travail uniquement pour la durée de l'appel. Si l'appel aboutit, le gestionnaire de files d'attente valide automatiquement l'unité d'oeuvre (l'application n'a pas besoin de le faire). Si l'appel échoue, le gestionnaire de files d'attente annule l'unité d'oeuvre.
- Si l'appel fonctionne en dehors d'une unité de travail définie par l'utilisateur, mais qu'une unité de travail définie par l'utilisateur existe, le gestionnaire de files d'attente ne peut pas effectuer de réassemblage. Si le message ne nécessite pas de réassemblage, l'appel peut tout de même aboutir. Mais si le message nécessite un réassemblage, l'appel échoue avec le code anomalie RC2255 .

Pour les messages non persistants, le gestionnaire de files d'attente n'a pas besoin qu'une unité d'oeuvre soit disponible pour effectuer le réassemblage.

Chaque message physique qui est un segment possède son propre descripteur de message. Pour les segments constituant un message logique unique, la plupart des zones du descripteur de message sont identiques pour tous les segments du message logique-en général, ce sont uniquement les zones *MDMID*, *MDOFF* et *MDMFL* qui diffèrent entre les segments du message logique. Toutefois, si un segment est placé dans une file d'attente de rebut sur un gestionnaire de files d'attente intermédiaire, le gestionnaire DLQ extrait le message spécifiant l'option *GMCONV*, ce qui peut entraîner la modification du jeu de caractères ou du codage du segment. Si le gestionnaire de file d'attente des messages non livrés envoie le segment en cours de route, le segment peut avoir un jeu de caractères ou un codage différent des autres segments du message logique lorsque le segment arrive finalement au gestionnaire de files d'attente de destination.

Un message logique composé de segments dans lesquels *MDCSI*, *MDENC* ou les deux zones diffèrent ne peut pas être réassemblé par le gestionnaire de files d'attente en un seul message logique. A la place, le gestionnaire de files d'attente réassemble et renvoie les premiers segments consécutifs au début du message logique ayant les mêmes identificateurs et codages de jeu de caractères, et l'appel *MQGET* se termine avec le code achèvement *CCWARN* et le code anomalie *RC2243* ou *RC2244* , selon le cas. Cela se produit que *GMCONV* soit spécifié ou non. Pour extraire les segments restants, l'application doit émettre à nouveau l'appel *MQGET* sans l'option *GMCMPM*, en extrayant les segments un par un. *GMLOGO* peut être utilisé pour extraire les segments restants dans l'ordre.

Il est également possible pour une application qui place des segments pour définir d'autres champs dans le descripteur de message à des valeurs différentes entre les segments. Cependant, cela ne présente aucun avantage si l'application de réception utilise *GMCMPM* pour extraire le message logique. Lorsque le gestionnaire de files d'attente réassemble un message logique, il renvoie dans le descripteur de message les valeurs du descripteur de message pour le premier segment ; la seule exception est la zone *MDMFL* , que le gestionnaire de files d'attente définit pour indiquer que le message réassemblé est le seul segment.

Si *GMCMPM* est spécifié pour un message de rapport, le gestionnaire de files d'attente effectue un traitement spécial. Le gestionnaire de files d'attente vérifie la file d'attente pour voir si tous les messages de rapport de ce type relatifs aux différents segments du message logique sont présents dans la file d'attente. Si tel est le cas, ils peuvent être extraits en tant que message unique en spécifiant *GMCMPM*. Pour que cela soit possible, les messages de rapport doivent être générés par un gestionnaire de files d'attente ou un agent MCA prenant en charge la segmentation, ou l'application d'origine doit demander au moins 100 octets de données de message (c'est-à-dire que les options *RO\* D* ou *RO\* F* appropriées doivent être spécifiées). Si la quantité de données d'application est inférieure à la quantité totale pour un segment, les octets manquants sont remplacés par des valeurs nulles dans le message de rapport renvoyé.

Si *GMCMPM* est spécifié avec *GMMUC* ou *GMBRWC*, le curseur de navigation doit être positionné sur un message avec une zone *MDOFF* dans *MQMD* dont la valeur est 0. Si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie *RC2246* .



GMCMPPM implique GMASGA, qui n'a donc pas besoin d'être spécifié.

GMCMPPM peut être spécifié avec n'importe laquelle des autres options GM\* à l'exception de GMPSYP, et avec n'importe laquelle des options MO\* à l'exception de MOOFFS.

### GMAMSA

Tous les messages du groupe doivent être disponibles.

Cette option indique que les messages d'un groupe ne peuvent être extraits que lorsque tous les messages du groupe sont disponibles. Si la file d'attente contient des groupes de messages dont certains des messages sont manquants (peut-être parce qu'ils ont été retardés sur le réseau et qu'ils ne sont pas encore arrivés), la spécification de GMAMSA empêche l'extraction des messages appartenant à des groupes incomplets. Toutefois, ces messages contribuent toujours à la valeur de l'attribut de file d'attente **CurrentQDepth** ; cela signifie qu'il peut n'y avoir aucun groupe de messages pouvant être extrait, même si **CurrentQDepth** est supérieur à zéro. Si aucun autre message n'est récupérable, le code raison RC2033 est renvoyé après l'expiration de l'intervalle d'attente indiqué (le cas échéant).

Le traitement de GMAMSA dépend de la spécification ou non de GMLOGO:

- Si les deux options sont spécifiées, GMAMSA n'affecte que lorsqu'il n'y a pas de groupe en cours ou de message logique. S'il existe un groupe ou un message logique en cours, GMAMSA est ignoré. Cela signifie que GMAMSA peut rester en fonction lors du traitement des messages dans l'ordre logique.
- Si GMAMSA est spécifié sans GMLOGO, GMAMSA a toujours un effet. Cela signifie que l'option doit être désactivée une fois que le premier message du groupe a été supprimé de la file d'attente, afin de pouvoir supprimer les messages restants du groupe.

L'exécution réussie d'un appel MQGET spécifiant GMAMSA signifie qu'au moment où l'appel MQGET a été émis, tous les messages du groupe se trouvaient dans la file d'attente. Toutefois, sachez que d'autres applications peuvent toujours supprimer des messages du groupe (le groupe n'est pas verrouillé pour l'application qui extrait le premier message du groupe).

Si cette option n'est pas spécifiée, les messages appartenant à des groupes peuvent être extraits même lorsque le groupe est incomplet.

GMAMSA implique GMASGA, qui n'a donc pas besoin d'être spécifié.

GMAMSA peut être spécifié avec l'une des autres options GM\* et avec l'une des options MO\*.

### GMASGA

Tous les segments d'un message logique doivent être disponibles.

Cette option indique que les segments d'un message logique ne peuvent être extraits que lorsque tous les segments du message logique sont disponibles. Si la file d'attente contient des messages segmentés dont certains des segments sont manquants (peut-être parce qu'ils ont été retardés sur le réseau et qu'ils ne sont pas encore arrivés), la spécification de GMASGA empêche l'extraction de segments appartenant à des messages logiques incomplets. Toutefois, ces segments contribuent toujours à la valeur de l'attribut de file d'attente **CurrentQDepth** ; cela signifie qu'il n'y a peut-être pas de messages logiques pouvant être extraits, même si **CurrentQDepth** est supérieur à zéro. Si aucun autre message n'est récupérable, le code raison RC2033 est renvoyé après l'expiration de l'intervalle d'attente indiqué (le cas échéant).

Le traitement de GMASGA dépend de la spécification ou non de GMLOGO:

- Si les deux options sont spécifiées, GMASGA n'a d'effet que lorsqu'il n'y a pas de message logique en cours. S'il existe un message logique en cours, GMASGA est ignoré. Cela signifie que GMASGA peut rester en fonction lors du traitement des messages dans l'ordre logique.
- Si GMASGA est spécifié sans GMLOGO, GMASGA a toujours un effet. Cela signifie que l'option doit être désactivée une fois que le premier segment du message logique a été supprimé de la file d'attente, afin de pouvoir supprimer les segments restants du message logique.

Si cette option n'est pas spécifiée, les segments de message peuvent être extraits même lorsque le message logique est incomplet.

Alors que GMCMPM et GMASGA exigent que tous les segments soient disponibles avant que l'un d'entre eux puisse être récupéré, le premier renvoie le message complet, tandis que le second permet de récupérer les segments un par un.

Si GMASGA est spécifié pour un message de rapport, le gestionnaire de files d'attente effectue un traitement spécial. Le gestionnaire de files d'attente vérifie la file d'attente pour voir s'il existe au moins un message de rapport pour chacun des segments qui constituent le message logique complet. Si tel est le cas, la condition GMASGA est satisfaite. Toutefois, le gestionnaire de files d'attente ne vérifie pas le type des messages de rapport présents et il peut donc y avoir une combinaison de types de rapport dans les messages de rapport relatifs aux segments du message logique. Par conséquent, le succès de GMASGA n'implique pas que GMCMPM réussisse. S'il existe une combinaison de types de rapport pour les segments d'un message logique particulier, ces messages de rapport doivent être extraits un par un.

GMASGA peut être spécifié avec l'une des autres options GM\* et avec l'une des options MO\*.

**Option par défaut:** Si aucune des options décrites n'est requise, l'option suivante peut être utilisée:

#### **MGMNONE**

Aucune option spécifiée.

Cette valeur peut être utilisée pour indiquer qu'aucune autre option n'a été spécifiée ; toutes les options prennent leurs valeurs par défaut. GMNONE est défini pour faciliter la documentation du programme ; il n'est pas prévu que cette option soit utilisée avec d'autres, mais comme sa valeur est zéro, une telle utilisation ne peut pas être détectée.

La valeur initiale de la zone *GMOPT* est GMNWT.

#### **GMRE1 (chaîne de caractères d'1 octet)**

Réservé.

Il s'agit d'une zone réservée. La valeur initiale de cette zone est un caractère blanc. Cette zone est ignorée si *GMVER* est inférieur à GMVER2.

#### **GMRL (entier signé à 10 chiffres)**

Longueur des données de message renvoyées (octets).

Il s'agit d'une zone de sortie définie par le gestionnaire de files d'attente sur la longueur en octets des données de message renvoyées par l'appel MQGET dans le paramètre **BUFFER** . Si le gestionnaire de files d'attente ne prend pas en charge cette fonction, *GMRL* est défini sur la valeur RLUNDF.

Lorsque des messages sont convertis entre des codages ou des jeux de caractères, les données de message peuvent parfois changer de taille. Au retour de l'appel MQGET:

- Si *GMRL* n'est pas RLUNDF, le nombre d'octets de données de message renvoyés est indiqué par *GMRL* .
- Si *GMRL* a la valeur RLUNDF, le nombre d'octets de données de message renvoyés est généralement indiqué par la valeur la plus faible ( *BUFLen* et *DATLEN* ), mais peut être inférieur à cette valeur si l'appel MQGET se termine avec le code anomalie RC2079 . Dans ce cas, les octets non significatifs du paramètre **BUFFER** sont définis sur des valeurs nulles.

La valeur spéciale suivante est définie:

#### **RLUNDF**

Longueur des données renvoyées non définie.

La valeur initiale de cette zone est RLUNDF. Cette zone est ignorée si *GMVER* est inférieur à GMVER3.

#### **GMRQN (chaîne de caractères de 48 octets)**

Nom résolu de la file d'attente de destination.

Il s'agit d'une zone de sortie définie par le gestionnaire de files d'attente sur le nom local de la file d'attente à partir de laquelle le message a été extrait, comme défini sur le gestionnaire de files d'attente local. Ce nom est différent du nom utilisé pour ouvrir la file d'attente si:

- Une file d'attente alias a été ouverte (auquel cas, le nom de la file d'attente locale dans laquelle l'alias a été résolu est renvoyé), ou
- Une file d'attente modèle a été ouverte (auquel cas, le nom de la file d'attente locale dynamique est renvoyé).

La longueur de cette zone est indiquée par LNQN. La valeur initiale de cette zone est de 48 caractères blancs.

#### **GMR52 (chaîne de caractères d'1 octet)**

Réservé.

Il s'agit d'une zone réservée. La valeur initiale de cette zone est un caractère blanc. Cette zone est ignorée si *GMVER* est inférieur à *GMVER4*.

#### **GMSEG (chaîne de caractères de 1 octet)**

Indicateur signalant si une segmentation supplémentaire est autorisée pour le message extrait.

Elle a l'une des valeurs suivantes :

##### **SEGIHB**

Segmentation non autorisée.

##### **SEGALW**

Segmentation autorisée.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est SEGIHB. Cette zone est ignorée si *GMVER* est inférieur à *GMVER2*.

#### **GMSG1 (entier signé à 10 chiffres)**

Signal.

Il s'agit d'une zone réservée ; sa valeur n'est pas significative. La valeur initiale de cette zone est 0.

#### **GMSG2 (entier signé à 10 chiffres)**

Identificateur de signal.

Il s'agit d'une zone réservée ; sa valeur n'est pas significative.

#### **GMSID (chaîne de caractères de 4 octets)**

Identificateur de structure.

La valeur doit être:

##### **GMSIDV**

Identificateur de la structure des options get-message.

Cette zone est toujours une zone d'entrée. La valeur initiale de cette zone est GMSIDV.

#### **GMSST (chaîne de caractères d'1 octet)**

Indicateur signalant si le message extrait est un segment d'un message logique.

Elle a l'une des valeurs suivantes :

##### **SSNSEG**

Le message n'est pas un segment.

##### **SSSEG**

Le message est un segment, mais n'est pas le dernier segment du message logique.

##### **SSLSEG**

Le message est le dernier segment du message logique.

Il s'agit également de la valeur renvoyée si le message logique se compose d'un seul segment.

Cette zone est une zone de sortie. La valeur initiale de cette zone est SSNSEG. Cette zone est ignorée si *GMVER* est inférieur à *GMVER2*.

### **GMTOK (chaîne de 16 octets)**

Jeton de message.

Il s'agit d'une zone réservée ; sa valeur n'est pas significative. La valeur spéciale suivante est définie:

#### **MTKNON**

Aucun jeton de message.

La valeur est zéro binaire pour la longueur de la zone.

La longueur de cette zone est indiquée par LNMTOK. La valeur initiale de cette zone est MTKNON. Cette zone est ignorée si *GMVER* est inférieur à GMVER3.

### **GMVER (entier signé à 10 chiffres)**

Numéro de version de la structure.

Il doit s'agir de l'une des valeurs suivantes :

#### **GMVER1**

Structure des options get-message Version-1 .

#### **GMVER2**

Structure des options d'obtention de message Version-2 .

#### **GMVER3**

Structure des options get-message Version-3 .

#### **GMVER4**

Structure des options get-message Version-4 .

Les zones qui existent uniquement dans les versions plus récentes de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

#### **GMVERC**

Version actuelle de la structure des options get-message.

Cette zone est toujours une zone d'entrée. La valeur initiale de cette zone est GMVER1.

### **GMVER (entier signé à 10 chiffres)**

Numéro de version de la structure.

Il doit s'agir de l'une des valeurs suivantes :

#### **GMVER1**

Structure des options get-message Version-1 .

#### **GMVER2**

Structure des options d'obtention de message Version-2 .

#### **GMVER3**

Structure des options get-message Version-3 .

#### **GMVER4**

Structure des options get-message Version-4 .

Les zones qui existent uniquement dans les versions plus récentes de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

#### **GMVERC**

Version actuelle de la structure des options get-message.

Cette zone est toujours une zone d'entrée. La valeur initiale de cette zone est GMVER1.

### **GMWI (entier signé à 10 chiffres)**

Intervalle d'attente.

Il s'agit de la durée approximative, exprimée en millisecondes, pendant laquelle l'appel MQGET attend l'arrivée d'un message approprié (c'est-à-dire un message répondant aux critères de sélection spécifiés dans le paramètre **MSGDSC** de l'appel MQGET ; voir la zone *MDMID* décrite dans «MQMD (descripteur de message) sous IBM i», à la page 1162 pour plus de détails). Si aucun message approprié n'est arrivé après ce délai, l'appel se termine avec le code CCFAIL et le code anomalie RC2033.

*GMWI* est utilisé avec l'option GMWT. Elle est ignorée si cette option n'est pas spécifiée. S'il est spécifié, *GMWI* doit être supérieur ou égal à zéro, ou la valeur spéciale suivante:

**WIULIM**

Intervalle d'attente illimité.

La valeur initiale de cette zone est 0.

**Valeurs initiales**

Tableau 704. Zones dans MQGMO

Nom de zone	Nom de la constante	Valeur de la constante
<i>GMSID</i>	GMSIDV	'GMO-'
<i>GMVER</i>	GMVER1	1
<i>GMOPT</i>	GMNWT	0
<i>GMWI</i>	Aucun	0
<i>GMSG1</i>	Aucun	0
<i>GMSG2</i>	Aucun	0
<i>GMRQN</i>	Aucun	Espaces vides
<i>GMMO</i>	MOMSGI + MOCORI	3
<i>GMGST</i>	GSNIG	' '
<i>GMSST</i>	SSNSEG	' '
<i>GMSEG</i>	SEGIHB	' '
<i>GMRE1</i>	Aucun	' '
<i>GMTOK</i>	MTKNON	Valeurs NULL
<i>GMRL</i>	RLUNDF	-1
<i>GMRS2</i>	Aucun	' '
<i>GMMH</i>	HNEANT	0

**Remarques :**

1. Le symbole - représente un caractère blanc unique.

**Déclaration RPG**

```
D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQGMO Structure
D*
D* Structure identifier
D GMSID          1      4    INZ('GMO ')
D* Structure version number
D GMVER          5      8I 0 INZ(1)
D* Options that control the action ofMQGET
```

```

D  GMOPT                9      12I 0 INZ(0)
D* Wait interval
D  GMWI                 13     16I 0 INZ(0)
D* Signal
D  GMSG1                17     20I 0 INZ(0)
D* Signal identifier
D  GMSG2                21     24I 0 INZ(0)
D* Resolved name of destination queue
D  GMRQN                25      72   INZ
D* Options controlling selection criteria used for MQGET
D  GMMO                 73     76I 0 INZ(3)
D* Flag indicating whether messageretrieved is in a group
D  GMGST                77      77   INZ(' ')
D* Flag indicating whether messageretrieved is a segment of a
D* logicalmessage
D  GMSST                78      78   INZ(' ')
D* Flag indicating whether furthersegmentation is allowed for themessage
D* retrieved
D  GMSEG                79      79   INZ(' ')
D* Reserved
D  GMRE1                80      80   INZ
D* Message token
D  GMTOK                81      96   INZ(X'0000000000000000-
D                          0000000000000000')
D* Length of message data returned(bytes)
D  GMRL                 97     100I 0 INZ(-1)
D* Reserved
D  GMRS2               101     104I 0 INZ(0)
D* Message handle
D  GMMH                 105     112I 0 INZ(0)

```

## IBM i MQIIH (en-tête d'informationIMS ) sous IBM i

La structure MQIIH décrit les informations qui doivent être présentes au début d'un message envoyé à la passerelle IMS via IBM MQ for z/OS.

### Présentation

**Nom de format:** FMIMS.

**Jeu de caractères et codage:** des conditions spéciales s'appliquent au jeu de caractères et au codage utilisés pour la structure MQIIH et les données de message d'application:

- Les applications qui se connectent au gestionnaire de files d'attente qui possède la file d'attente de pont IMS doivent fournir une structure MQIIH qui se trouve dans le jeu de caractères et le codage du gestionnaire de files d'attente. En effet, la conversion de données de la structure MQIIH n'est pas effectuée dans ce cas.
- Les applications qui se connectent à d'autres gestionnaires de files d'attente peuvent fournir une structure MQIIH qui se trouve dans l'un des jeux de caractères et codages pris en charge ; la conversion de MQIIH est effectuée par l'agent de canal de réception des messages connecté au gestionnaire de files d'attente qui possède la file d'attente de pont IMS .

**Remarque :** Il y a une exception à cela. Si le gestionnaire de files d'attente propriétaire de la file d'attente de pont IMS utilise CICS pour la mise en file d'attente répartie, le MQIIH doit se trouver dans le jeu de caractères et le codage du gestionnaire de files d'attente propriétaire de la file d'attente de pont IMS .

- Les données de message d'application qui suivent la structure MQIIH doivent être dans le même jeu de caractères et le même codage que la structure MQIIH. Les zones *IICSI* et *IIENC* de la structure MQIIH ne peuvent pas être utilisées pour spécifier le jeu de caractères et le codage des données de message d'application.

Un exit de conversion de données doit être fourni par l'utilisateur pour convertir les données de message d'application si les données ne sont pas l'un des formats intégrés pris en charge par le gestionnaire de files d'attente.

- [«Authentification des passtickets pour les applications de pont IMS»](#), à la page 1151
- [«Zones»](#), à la page 1151

- «Valeurs initiales», à la page 1154
- «Déclaration RPG», à la page 1154

## Authentification des passtickets pour les applications de pont IMS

Il est désormais possible pour les administrateurs IBM MQ de spécifier le nom d'application à utiliser pour l'authentification des tickets d'accès, pour les applications de pont IMS . Pour ce faire, le nom d'application est indiqué en tant que nouvel attribut PTKTAPPL pour la définition d'objet STGCLASS, sous la forme d'une chaîne alphanumérique de 1 à 8 caractères.

Une valeur vide signifie que l'authentification a lieu comme avec les versions précédentes de IBM MQ, c'est-à-dire qu'aucun nom d'application ne circule dans la demande d'authentification et que la valeur MVSxxxx est utilisée à la place.

Une valeur de 1 à 8 caractères alphanumériques doit respecter les règles relatives aux noms d'application Passticket, comme décrit dans les publications RACF .

Les administrateurs IBM MQ et RACF doivent convenir des noms d'application valides à utiliser. L'administrateur RACF doit créer un profil dans la classe PTKTDATA qui accorde un accès en lecture (READ) aux ID utilisateur de toutes les applications auxquelles l'accès doit être accordé. L'administrateur IBM MQ doit créer ou modifier les définitions STGCLASS requises qui indiquent le nom d'application à utiliser pour l'authentification par mot de passe.

Pour plus d'informations, voir *Script (MQSC) Command Reference*.

## Zones

La structure MQIIH contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

### IIAUT (chaîne de caractères de 8 octets)

Mot de passe ou mot de passe RACF .

Ce paramètre est facultatif ; s'il est spécifié, il est utilisé avec l'ID utilisateur dans le contexte de sécurité MQMD pour générer un UTOKEN envoyé à IMS afin de fournir un contexte de sécurité. S'il n'est pas spécifié, l'ID utilisateur est utilisé sans vérification. Cela dépend des paramètres des commutateurs RACF , qui peuvent nécessiter la présence d'un authentificateur.

Cette valeur est ignorée si le premier octet est vide ou null. La valeur spéciale suivante peut être utilisée:

#### IAUNON

Pas d'authentification.

La longueur de cette zone est indiquée par LNAUTH. La valeur initiale de cette zone est IAUNON.

### IICMT (chaîne de caractères de 1 octet)

Mode de validation.

Pour plus d'informations sur les modes de validation IMS , voir *OTMA Reference* . Il doit s'agir de l'une des valeurs suivantes :

#### ICMCTS

Validez puis envoyez.

Ce mode implique une double mise en file d'attente de la sortie, mais des temps d'occupation de région plus courts. Les transactions rapides et conversationnelles ne peuvent pas s'exécuter avec ce mode.

#### ICMSTC

Envoyer puis valider.

La valeur initiale de cette zone est ICMCTS.

**IICSI (entier signé à 10 chiffres)**

Réservé.

Il s'agit d'une zone réservée ; sa valeur n'est pas significative. La valeur initiale de cette zone est 0.

**IIENC (entier signé à 10 chiffres)**

Réservé.

Il s'agit d'une zone réservée ; sa valeur n'est pas significative. La valeur initiale de cette zone est 0.

**IIFLG (entier signé à 10 chiffres)**

Indicateurs.

La valeur doit être:

**IINONE**

Aucun indicateur.

La valeur initiale de cette zone est IINONE.

**IIFMT (chaîne de caractères de 8 octets)**

Nom de format IBM MQ des données qui suivent MQIIH.

Indique le nom de format IBM MQ des données qui suivent la structure MQIIH.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de la zone *MDFMT* dans MQMD.

La longueur de cette zone est indiquée par LNFMT. La valeur initiale de cette zone est FMNONE.

**IILEN (entier signé à 10 chiffres)**

Longueur de la structure MQIIH.

La valeur doit être:

**IILEN1**

Longueur de la structure d'en-tête d'informations IMS.

La valeur initiale de cette zone est IILEN1.

**IILTO (chaîne de caractères de 8 octets)**

Remplacement de terminal logique.

Il est placé dans le champ IO PCB. Elle est facultative ; si elle n'est pas spécifiée, le nom TPIPE est utilisé. Elle est ignorée si le premier octet est vide ou null.

La longueur de cette zone est donnée par LNLTOV. La valeur initiale de cette zone est de 8 caractères blancs.

**IIMMN (chaîne de caractères de 8 octets)**

Nom de mappe des services de format de message.

Il est placé dans le champ IO PCB. Il est facultatif. En entrée, il représente le MID, en sortie, il représente le MOD. Il est ignoré si le premier octet est vide ou nul.

La longueur de cette zone est indiquée par LNMFMN. La valeur initiale de cette zone est de 8 caractères blancs.

**IIRFM (chaîne de caractères de 8 octets)**

Nom de format IBM MQ du message de réponse.

Il s'agit du nom de format IBM MQ du message de réponse qui sera envoyé en réponse au message en cours. Les règles de codage sont les mêmes que celles de la zone *MDFMT* dans MQMD.

La longueur de cette zone est indiquée par LNFMT. La valeur initiale de cette zone est FMNONE.



### **IIRSV (chaîne de caractères de 1 octet)**

Réservé.

Il s'agit d'une zone réservée ; elle doit être vide.

### **IISEC (chaîne de caractères de 1 octet)**

Portée de la sécurité.

Indique le traitement de sécurité IMS requis. Les valeurs suivantes sont définies :

#### **ISSCHK**

Vérifiez la portée de la sécurité.

Un ACEE est généré dans la région de contrôle, mais pas dans la région dépendante.

#### **ISSFUL**

Portée de sécurité complète.

Un environnement ACEE mis en cache est généré dans la région de contrôle et un environnement ACEE non mis en cache est généré dans la région dépendante. Si vous utilisez ISSFUL, vous devez vous assurer que l'ID utilisateur pour lequel ACEE est généré a accès aux ressources utilisées dans la région dépendante.

Si ISSCHK et ISSFUL ne sont pas indiqués pour cette zone, ISSCHK est utilisé par défaut.

La valeur initiale de cette zone est ISSCHK.

### **IISID (chaîne de caractères de 4 octets)**

Identificateur de structure.

La valeur doit être:

#### **IISIDV**

Identificateur de la structure d'en-tête d'informations IMS .

La valeur initiale de cette zone est IISIDV.

### **IITID (chaîne de 16 octets)**

Identificateur de l'instance de transaction.

Cette zone étant utilisée par les messages de sortie de IMS , elle est ignorée lors de la première entrée. Si *IITST* est défini sur ITSIC, il doit être fourni dans l'entrée suivante, ainsi que dans toutes les entrées suivantes, pour permettre à IMS de corréler les messages à la conversation correcte. La valeur spéciale suivante peut être utilisée:

#### **NON ITIE**

Aucun ID d'instance de transaction.

La longueur de cette zone est indiquée par LNTIID. La valeur initiale de cette zone est ITINON.

### **IITST (chaîne de caractères de 1 octet)**

Etat de la transaction.

Indique l'état de la conversation IMS . Elle est ignorée lors de la première entrée car aucune conversation n'existe. Sur les entrées suivantes, il indique si une conversation est active ou non. En sortie, il est défini par IMS. Il doit s'agir de l'une des valeurs suivantes :

#### **ITSIC**

En conversation.

#### **ITSNIC**

Pas en conversation.

#### **ITSARC**

Renvoie les données d'état de transaction sous forme structurée.

Cette valeur est utilisée uniquement avec la commande IMS /DISPLAY TRAN . Elle entraîne le renvoi des données d'état de la transaction dans la forme structurée IMS au lieu de la forme alphanumérique. Pour plus d'informations, voir Ecriture de programmes de transaction IMS via IBM MQ .

La valeur initiale de cette zone est ITSNIC.

### IIVER (entier signé à 10 chiffres)

Numéro de version de la structure.

La valeur doit être:

#### IIVER1

Numéro de version de la structure d'en-tête d'informations IMS .

La constante suivante indique le numéro de version de la version en cours:

#### IiOS

Version actuelle de la structure d'en-tête d'informations IMS .

La valeur initiale de cette zone est IIVER1.

## Valeurs initiales

Tableau 705. Zones dans MQIIH		
Nom de zone	Nom de la constante	Valeur de la constante
IISID	IISIDV	' IIH↵ '
IIVER	IIVER1	1
IILEN	IILEN1	84
IIENC	Aucun	0
IICSI	Aucun	0
IIFMT	FMAUCUN	Espaces vides
IIFLG	IINONE	0
IILTO	Aucun	Espaces vides
IIMMN	Aucun	Espaces vides
IIRFM	FMAUCUN	Espaces vides
IIAUT	IAUNON	Espaces vides
IITID	NON ITIE	Valeurs NULL
IITST	ITSNIC	' '
IICMT	ICMCTS	' 0 '
IISEC	ISSCHK	' C '
IIRSV	Aucun	' '

### Remarques :

1. Le symbole ↵ représente un caractère blanc unique.

## Déclaration RPG

D\*..1.....2.....3.....4.....5.....6.....7..

```

D*
D* MQIIH Structure
D*
D* Structure identifier
D IISID 1 4 INZ('IIH ')
D* Structure version number
D IIVER 5 8I 0 INZ(1)
D* Length of MQIIH structure
D IILEN 9 12I 0 INZ(84)
D* Reserved
D IIENC 13 16I 0 INZ(0)
D* Reserved
D IICSI 17 20I 0 INZ(0)
D* MQ format name of data that followsMQIIH
D IIFMT 21 28 INZ(' ')
D* Flags
D IIFLG 29 32I 0 INZ(0)
D* Logical terminal override
D IILTO 33 40 INZ
D* Message format services map name
D IIMMN 41 48 INZ
D* MQ format name of reply message
D IIRFM 49 56 INZ(' ')
D* RACF password or passticket
D IIAUT 57 64 INZ(' ')
D* Transaction instance identifier
D IITID 65 80 INZ(X'0000000000000000-
D 0000000000000000')
D* Transaction state
D IITST 81 81 INZ(' ')
D* Commit mode
D IICMT 82 82 INZ('0')
D* Security scope
D IISEC 83 83 INZ('C')
D* Reserved
D IIRSV 84 84 INZ

```

## MQIMPO (Inquire message property options) sur IBM i

La structure MQIMPO permet aux applications de spécifier des options qui contrôlent la manière dont les propriétés des messages sont recherchées.

### Présentation

**Objectif:** la structure est un paramètre d'entrée dans l'appel MQINQMP.

**Jeu de caractères et codage:** les données de MQIMPO doivent être dans le jeu de caractères de l'application et le codage de l'application (ENNAT).

- [«Zones», à la page 1155](#)
- [«Valeurs initiales», à la page 1161](#)
- [«Déclaration RPG», à la page 1161](#)

### Zones

La structure MQIMPO contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

#### IPOPT (entier signé à 10 chiffres)

Les options suivantes contrôlent l'action de MQINQMP. Vous pouvez spécifier une ou plusieurs de ces options. Pour spécifier plusieurs options, ajoutez les valeurs ensemble (n'ajoutez pas la même constante plusieurs fois) ou combinez les valeurs à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations bit). Les combinaisons d'options non valides sont notées ; toutes les autres combinaisons sont valides.

**Options de données de valeur:** Les options suivantes concernent le traitement des données de valeur lorsque la propriété est extraite du message.

## VALIPCAL

Cette option demande que la valeur de la propriété soit convertie pour être conforme aux valeurs *IPREQCSI* et *IPREQENC* spécifiées avant que l'appel MQINQMP ne renvoie la valeur de la propriété dans la zone *Value* .

- Si la conversion aboutit, les zones *IPRETCSI* et *IPRETENC* sont définies sur les mêmes valeurs que *IPREQCSI* et *IPREQENC* en cas de retour de l'appel MQINQMP.
- Si la conversion échoue, mais que l'appel MQINQMP se termine sans erreur, la valeur de la propriété est renvoyée non convertie.

Si la propriété est une chaîne, les zones *IPRETCSI* et *IPRETENC* sont définies sur le jeu de caractères et le codage de la chaîne non convertie.

Le code achèvement est CCWARN dans ce cas, avec le code anomalie RC2466. Le curseur de propriété est avancé vers la propriété renvoyée.

Si la valeur de la propriété se développe lors de la conversion et dépasse la taille du paramètre **Value** , la valeur est renvoyée non convertie, avec le code achèvement CCFAIL ; le code anomalie est défini sur RC2469.

Le paramètre **DataLength** de l'appel MQINQMP renvoie la longueur à laquelle la valeur de propriété aurait été convertie, afin de permettre à l'application de déterminer la taille de la mémoire tampon requise pour prendre en charge la valeur de propriété convertie. Le curseur de propriété est inchangé.

Cette option demande également que:

- Si le nom de la propriété contient un caractère générique, et
- La zone *IPRETNAMECHRP* est initialisée avec une adresse ou un décalage pour le nom renvoyé, le nom renvoyé est converti pour être conforme aux valeurs *IPREQCSI* et *IPREQENC* .
- Si la conversion aboutit, la zone *VSCCSID* de *IPRETNAMECHRP* et le codage du nom renvoyé sont définis sur la valeur d'entrée *IPREQCSI* et *IPREQENC* .
- Si la conversion échoue, mais que l'appel MQINQMP se termine sans erreur ni avertissement, le nom renvoyé n'est pas converti. Le code achèvement est CCWARN dans ce cas, avec le code anomalie RC2492.

Le curseur de propriété est avancé vers la propriété renvoyée. RC2466 est renvoyé si la valeur et le nom ne sont pas convertis.

Si le nom renvoyé se développe lors de la conversion et dépasse la taille de la zone *VSBufsize* de la *RequestedName*, la chaîne renvoyée reste non convertie, avec le code achèvement CCFAIL et le code anomalie RC2465.

La zone *VSLength* de la structure MQCHARV renvoie la longueur à laquelle la valeur de propriété aurait été convertie, afin de permettre à l'application de déterminer la taille de la mémoire tampon requise pour accueillir la valeur de propriété convertie. Le curseur de propriété est inchangé.

## IPCTYP

Cette option demande que la valeur de la propriété soit convertie de son type de données en cours en type de données spécifié dans le paramètre **Type** de l'appel MQINQMP.

- Si la conversion aboutit, le paramètre **Type** est inchangé lors du renvoi de l'appel MQINQMP.
- Si la conversion échoue, mais que l'appel MQINQMP se termine sans erreur, l'appel échoue avec la raison RC2470. Le curseur de propriété est inchangé.

Si la conversion du type de données entraîne l'extension de la valeur lors de la conversion et que la valeur convertie dépasse la taille du paramètre **Value** , la valeur est renvoyée non convertie, avec le code achèvement CCFAIL et le code anomalie RC2469.

Le paramètre **DataLength** de l'appel MQINQMP renvoie la longueur à laquelle la valeur de propriété aurait été convertie, afin de permettre à l'application de déterminer la taille de la

mémoire tampon requise pour prendre en charge la valeur de propriété convertie. Le curseur de propriété est inchangé.

Si la valeur du paramètre **Type** de l'appel MQINQMP n'est pas valide, l'appel échoue avec le motif RC2473.

Si la conversion de type de données demandée n'est pas prise en charge, l'appel échoue avec le motif RC2470. Les conversions de type de données suivantes sont prises en charge:

<i>Tableau 706. Conversions de type de données prises en charge</i>	
Type de données de propriété	Types de données cible pris en charge
Connaissance de type	TYPSTR, TYPI8, TYPI16, TYPI32, TYPI64
TYPBST	TYPSTR
TYPI8	TYPSTR, TYPI16, TYPI32, TYPI64
TYPI16	TYPSTR, TYPI32, TYPI64
TYPI32	TYPSTR, TYPI64
TYPI64	TYPSTR
TYPF32	TYPSTR, TYPF64
TYPF64	TYPSTR
TYPSTR	TYPBOL, TYPI8, TYPI16, TYPI32, TYPI64, TYPF32, TYPF64
TYPNUL	Aucun

Les règles générales régissant les conversions prises en charge sont les suivantes:

- Les valeurs de propriété numérique peuvent être converties d'un type de données à un autre, à condition qu'aucune donnée ne soit perdue lors de la conversion.

Par exemple, la valeur d'une propriété avec le type de données TYPI32 peut être convertie en valeur avec le type de données TYPI64, mais ne peut pas être convertie en valeur avec le type de données TYPI16.

- Une valeur de propriété de tout type de données peut être convertie en chaîne.
- Une valeur de propriété de type chaîne peut être convertie en tout autre type de données, du moment que la chaîne est correctement formatée pour la conversion. Si une application tente de convertir une valeur de propriété de chaîne qui n'est pas formatée correctement, IBM MQ renvoie le code anomalie RC2472.
- Si une application tente une conversion qui n'est pas prise en charge, IBM MQ renvoie le code anomalie RC2470.

Les règles spécifiques pour la conversion d'une valeur de propriété d'un type de données à un autre sont les suivantes:

- Lors de la conversion d'une valeur de propriété TYPBOL en chaîne, la valeur TRUE est convertie en chaîne "TRUE" et la valeur false est convertie en chaîne "FALSE".
- Lors de la conversion d'une valeur de propriété TYPBOL en type de données numériques, la valeur TRUE est convertie en un et la valeur FALSE est convertie en zéro.
- Lors de la conversion d'une valeur de propriété de chaîne en valeur TYPBOL, la chaîne "TRUE", ou "1", est convertie en TRUE, et la chaîne "FALSE", ou "0", est convertie en FALSE.

Notez que les termes "TRUE" et "FALSE" ne sont pas sensibles à la casse.

Toute autre chaîne ne peut pas être convertie ; IBM MQ renvoie le code anomalie RC2472.

- Lors de la conversion d'une valeur de propriété de chaîne en valeur avec le type de données TYPI8, TYPI16, TYPI32 ou TYPI64, la chaîne doit avoir le format suivant:

```
[blanks][sign]digits
```

Les significations des composants de la chaîne sont les suivantes:

**blanks**

Caractères blancs de début facultatifs

**sign**

Caractère facultatif de signe plus (+) ou moins (-).

**digits**

Séquence contiguë de caractères numériques (0-9). Au moins un chiffre doit être présent.

Après la séquence de caractères numériques, la chaîne peut contenir d'autres caractères autres que des chiffres, mais la conversion s'arrête au premier caractère de ce type. La chaîne est supposée représenter un entier décimal.

IBM MQ renvoie le code anomalie RC2472 si la chaîne n'est pas formatée correctement.

- Lors de la conversion d'une valeur de propriété de chaîne en une valeur avec le type de données TYPF32 ou TYPF64, la chaîne doit avoir le format suivant:

```
[blanks][sign]digits[.digits][e_char[e_sign]e_digits]
```

Les significations des composants de la chaîne sont les suivantes:

**blanks**

Caractères blancs de début facultatifs

**sign**

Caractère facultatif de signe plus (+) ou moins (-).

**digits**

Séquence contiguë de caractères numériques (0-9). Au moins un chiffre doit être présent.

**e\_char**

Un caractère exposant, qui est soit "E", soit "e".

**e\_sign**

Signe plus (+) ou signe moins (-) facultatif pour l'exposant.

**e\_digits**

Séquence contiguë de caractères numériques (0-9) pour l'exposant. Au moins un chiffre doit être présent si la chaîne contient un caractère exposant.

Après la séquence de chiffres, la chaîne peut contenir d'autres caractères autres que des chiffres, mais la conversion s'arrête au premier caractère de ce type. La chaîne est supposée représenter un nombre décimal en virgule flottante avec un exposant puissance de 10.

IBM MQ renvoie le code anomalie RC2472 si la chaîne n'est pas formatée correctement.

- Lors de la conversion d'une valeur de propriété numérique en chaîne, la valeur est convertie en représentation de chaîne de la valeur sous forme de nombre décimal, et non en chaîne contenant le caractère ASCII pour cette valeur. Par exemple, l'entier 65 est converti en chaîne "65", et non en chaîne "A".
- Lors de la conversion d'une valeur de propriété de chaîne d'octets en chaîne, chaque octet est converti en deux caractères hexadécimaux représentant l'octet. Par exemple, le tableau d'octets {0xF1, 0x12, 0x00, 0xFF} est converti en chaîne "F11200FF".

**IQLEN**

Interrogez le type et la longueur de la valeur de propriété. La longueur est renvoyée dans le paramètre **DataLength** de l'appel MQINQMP. La valeur de la propriété n'est pas renvoyée.

Si une mémoire tampon *ReturnedName* est spécifiée, la zone *VSLength* de la structure MQCHARV est renseignée avec la longueur du nom de propriété. Le nom de propriété n'est pas renvoyé.

**Options d'itération:** Les options suivantes concernent l'itération sur les propriétés, à l'aide d'un nom avec un caractère générique

### **FIPINQF**

Renseignez-vous sur la première propriété qui correspond au nom spécifié. Après cet appel, un curseur est établi sur la propriété renvoyée.

Il s'agit de la valeur par défaut.

L'option IPINQC peut ensuite être utilisée avec un appel MQINQMP, si nécessaire, pour demander à nouveau la même propriété.

Notez qu'il n'y a qu'un seul curseur de propriété ; par conséquent, si le nom de propriété, spécifié dans l'appel MQINQMP, change le curseur est réinitialisé.

Cette option n'est pas valide avec l'une des options suivantes:

Nom de domaine de l'adresse IP  
IPINQC

### **Nom de domaine de l'adresse IP**

Demande sur la propriété suivante qui correspond au nom spécifié, en poursuivant la recherche à partir du curseur de propriété. Le curseur est avancé à la propriété qui est renvoyée.

S'il s'agit du premier appel MQINQMP pour le nom spécifié, la première propriété correspondant au nom spécifié est renvoyée.

L'option IPINQC peut ensuite être utilisée avec un appel MQINQMP si nécessaire, pour demander à nouveau la même propriété.

Si la propriété sous le curseur a été supprimée, MQINQMP renvoie la propriété correspondante suivante après celle qui a été supprimée.

Si une propriété qui correspond au caractère générique est ajoutée alors qu'une itération est en cours, elle peut ou non être renvoyée lors de la fin de l'itération. La propriété est renvoyée une fois que l'itération redémarre à l'aide d'IPINQF.

Une propriété correspondant au caractère générique qui a été supprimé alors que l'itération était en cours n'est pas renvoyée après sa suppression.

Cette option n'est pas valide avec l'une des options suivantes:

FIPINQF  
IPINQC

### **IPINQC**

Extrayez la valeur de la propriété pointée par le curseur de propriété. La propriété indiquée par le curseur de propriété est celle qui a été interrogée pour la dernière fois, à l'aide de l'option IPINQF ou IPINQN.

Le curseur de propriété est réinitialisé lorsque le descripteur de message est réutilisé, lorsque le descripteur de message est spécifié dans la zone *MsgHandle* du MQGMO sur un appel MQGET, ou lorsque le descripteur de message est spécifié dans les zones *OriginalMsgHandle* ou *NewMsgHandle* de la structure MQPMO sur un appel MQPUT.

Si cette option est utilisée lorsque le curseur de propriété n'a pas encore été établi, ou si la propriété pointée par le curseur de propriété a été supprimée, l'appel échoue avec le code achèvement CCFAIL et le motif RC2471.

Cette option n'est pas valide avec l'une des options suivantes:

FIPINQF  
Nom de domaine de l'adresse IP

Si aucune des options décrites précédemment n'est requise, l'option suivante peut être utilisée:

## **IPAUCUN**

Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. Toutes les options sont définies sur leur valeur par défaut.

IPNONE aide la documentation du programme ; il n'est pas prévu que cette option soit utilisée avec une autre option, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est IPINQF.

## **IPREQCSI (entier signé à 10 chiffres)**

Jeu de caractères dans lequel la valeur de la propriété recherchée doit être convertie si la valeur est une chaîne de caractères. Il s'agit également du jeu de caractères dans lequel le *ReturnedName* doit être converti lorsque IPCVAL ou IPCTYP est spécifié.

La valeur initiale de cette zone est CSAPL.

## **IPREQENC (entier signé à 10 chiffres)**

Il s'agit du codage dans lequel la valeur de la propriété interrogée doit être convertie lorsque IPCVAL ou IPCTYP est spécifié.

La valeur initiale de cette zone est ENNAT.

## **IPRE1 (entier signé à 10 chiffres)**

Il s'agit d'une zone réservée. La valeur initiale de cette zone est un caractère blanc.

## **IPRETCSI (entier signé à 10 chiffres)**

En sortie, il s'agit du jeu de caractères de la valeur renvoyée si le paramètre **Type** de l'appel MQINQMP est TYPSTR.

Si l'option IPCVAL est spécifiée et que la conversion a abouti, la zone *ReturnedCCSID*, en cas de retour, est identique à la valeur transmise.

La valeur initiale de cette zone est zéro.

## **IPRETENC (entier signé à 10 chiffres)**

En sortie, il s'agit du codage de la valeur renvoyée.

Si l'option IPCVAL est spécifiée et que la conversion a abouti, la zone *ReturnedEncoding*, en cas de retour, est identique à la valeur transmise.

La valeur initiale de cette zone est ENNAT.

## **IPRETNAMCHRP (entier signé à 10 chiffres)**

Nom réel de la propriété interrogée.

En entrée, une mémoire tampon de chaîne peut être transmise à l'aide de la zone *VSPtr* ou *VSOffset* de la structure MQCHARV. La longueur de la mémoire tampon de chaîne est spécifiée à l'aide de la zone *VSBuFSIZE* de la structure MQCHARV.

Lors du retour de l'appel MQINQMP, la mémoire tampon de la chaîne est terminée avec le nom de la propriété qui a été interrogée, à condition que la mémoire tampon de la chaîne soit suffisamment longue pour contenir complètement le nom. La zone *VSLength* de la structure MQCHARV est renseignée avec la longueur du nom de propriété. La zone *VSCCSID* de la structure MQCHARV est renseignée pour indiquer le jeu de caractères du nom renvoyé, que la conversion du nom ait échoué ou non.



Il s'agit d'une zone d'entrée/sortie. La valeur initiale de cette zone est MQCHARV\_DEFAULT.

### **IPSID (entier signé à 10 chiffres)**

Il s'agit de l'identificateur de structure. La valeur doit être:

#### **ISIDV**

Identificateur de la structure des options d'interrogation des propriétés de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est IPSIDV.

### **IPTYP (entier signé à 10 chiffres)**

Représentation sous forme de chaîne du type de données de la propriété.

Si la propriété a été spécifiée dans un en-tête MQRFH2 et que l'attribut MQRFH2 dt n'est pas reconnu, cette zone peut être utilisée pour déterminer le type de données de la propriété. *TypeString* est renvoyé dans le jeu de caractères codés 1208 (UTF-8) et correspond aux huit premiers octets de la valeur de l'attribut dt de la propriété dont la reconnaissance a échoué.

Il s'agit toujours d'une zone de sortie. La valeur initiale de cette zone est la chaîne nulle dans le langage de programmation C et 8 caractères blancs dans les autres langages de programmation.

### **IPVER (entier signé à 10 chiffres)**

Il s'agit du numéro de version de la structure. La valeur doit être:

#### **IPVER1**

Numéro de version de la structure des options de propriété de message d'interrogation.

La constante suivante indique le numéro de version de la version en cours:

#### **CVERIP**

Version actuelle de la structure des options de propriété de message d'interrogation.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est IPVER1.

## **Valeurs initiales**

<b>Nom de zone</b>	<b>Nom de la constante</b>	<b>Valeur de la constante</b>
<i>IPSID</i>	ISIDV	'IMPO'
<i>IPVER</i>	IPVER1	1
<i>IPOPT</i>	FIPINQF	
<i>IPREQENC</i>	ENNAT	
<i>IPREQCSI</i>	CSAPL	
<i>IPRETENC</i>	ENNAT	
<i>IPRETCSI</i>	0	
<i>IPRE1</i>	0	
<i>IPRETAMCHRP</i>		
<i>IPTYP</i>		blancs

## **Déclaration RPG**

D\* MQIMPO Structure  
D\*

```

D*
D* Structure identifier
D IPSID      1  4  INZ('IMPO')
D*
D* Structure version number
D IPVER      5  8I 0  INZ(1)
D*
** Options that control the action of
D* MQINQMP
D IPOPT      9  12I 0  INZ(0)
D*
D* Requested encoding of Value
D IPREQENC   13  16I 0  INZ(273)
D*
** Requested character set identifier
D* of Value
D IPREQCSI   17  20I 0  INZ(-3)
D*
D* Returned encoding of Value
D IPRETENC   21  24I 0  INZ(273)
D*
** Returned character set identifier of
D* Value
D IPRETCSI   25  28I 0  INZ(0)
D*
D* Reserved
D IPRE1      29  32I 0  INZ(0)
D*
D* Returned property name
D* Address of variable length string
D IPRETNAMCHRP 33  48*  INZ(*NULL)
D* Offset of variable length string
D IPRETNAMCHRO 49  52I 0  INZ(0)
D* Size of buffer
D IPRETNAMVSBS 53  56I 0  INZ(-1)
D* Length of variable length string
D IPRETNAMCHRL 57  60I 0  INZ(0)
D* CCSID of variable length string
D IPRETNAMCHRC 61  64I 0  INZ(-3)
D*
D* Property data type as a string
D IPTYP      65  72  INZ

```

## IBM i MQMD (descripteur de message) sous IBM i

### Présentation

**Objectif:** La structure MQMD contient les informations de contrôle qui accompagnent les données d'application lorsqu'un message circule entre les applications d'envoi et de réception. La structure est un paramètre d'entrée/sortie sur les appels MQGET, MQPUT et MQPUT1 .

**Versión:** la version actuelle de MQMD est MDVER2. Les zones qui existent uniquement dans les versions les plus récentes de la structure sont identifiées comme telles dans les descriptions qui suivent.

Le fichier COPY fourni contient la version la plus récente de MQMD prise en charge par l'environnement, mais avec la valeur initiale de la zone MDVER définie sur MDVER1. Pour utiliser des zones qui ne sont pas présentes dans la structure version-1 , l'application doit définir la zone MDVER sur le numéro de version de la version requise.

Une déclaration pour la structure version-1 est disponible avec le nom MQMD1.

**Jeu de caractères et codage:** les données dans MQMD doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par ENNAT. Toutefois, si l'application s'exécute en tant que IBM MQ MQI client, la structure doit être dans le jeu de caractères et le codage du client.

Si les gestionnaires de files d'attente d'envoi et de réception utilisent des jeux de caractères ou des codages différents, les données de MQMD sont converties automatiquement. Il n'est pas nécessaire que l'application convertisse le MQMD.

- [«Utilisation de différentes versions de MQMD», à la page 1163](#)

- «Contexte de message», à la page [1163](#)
- «Date d'expiration du message», à la page [1164](#)
- «Zones», à la page [1164](#)
- «Valeurs initiales», à la page [1207](#)
- «Déclaration RPG», à la page [1208](#)

## Utilisation de différentes versions de MQMD

Un MQMD version-2 est généralement équivalent à l'utilisation d'un MQMD version-1 et à la préfixation des données de message avec une structure MQMDE. Toutefois, si toutes les zones de la structure MQMDE ont leurs valeurs par défaut, MQMDE peut être omis. version-1 MQMD plus MQMDE sont utilisés comme décrit plus loin dans cette section.

- Dans les appels MQPUT et MQPUT1, si l'application fournit un MQMD version-1, l'application peut éventuellement préfixer les données de message avec un MQMDE, en définissant la zone MDFMT dans MQMD sur FMMDE pour indiquer qu'un MQMDE est présent. Si l'application ne fournit pas de MQMDE, le gestionnaire de files d'attente utilise les valeurs par défaut pour les zones du MQMDE.

**Remarque :** Plusieurs des zones qui existent dans le MQMD version-2 mais pas dans le MQMD version-1 sont des zones d'entrée/sortie sur les appels MQPUT et MQPUT1. Toutefois, le gestionnaire de files d'attente ne renvoie aucune valeur dans les zones équivalentes de MQMDE dans la sortie des appels MQPUT et MQPUT1 ; si l'application requiert ces valeurs de sortie, elle doit utiliser un MQMD version-2.

- Dans l'appel MQGET, si l'application fournit un MQMD version-1, le gestionnaire de files d'attente préfixe le message renvoyé avec un MQMDE, mais uniquement si une ou plusieurs des zones du MQMDE ont une valeur autre que la valeur par défaut. La zone MDFMT dans MQMD aura la valeur FMMDE pour indiquer qu'un MQMDE est présent.

Les valeurs par défaut utilisées par le gestionnaire de files d'attente pour les zones de MQMDE sont identiques aux valeurs initiales de ces zones, indiquées dans le [Tableau 709](#), à la page [1207](#).

Lorsqu'un message se trouve dans une file d'attente de transmission, certaines zones de MQMD sont définies sur des valeurs particulières ; pour plus de détails, voir «[MQXQH \(en-tête de file d'attente de transmission\) sous IBM i](#)», à la page [1306](#).

## Contexte de message

Certaines zones de MQMD contiennent le contexte de message. Généralement :

- Le *contexte d'identité* est lié à l'application qui a initialement inséré le message
- Le *contexte d'origine* est lié à l'application qui a inséré le message le plus récemment.
- Le *contexte utilisateur* est lié à l'application qui a initialement inséré le message.

Ces deux applications peuvent être la même application, mais elles peuvent également être des applications différentes (par exemple, lorsqu'un message est transmis d'une application à une autre).

Bien que le contexte d'identité et d'origine ait généralement la signification décrite précédemment, le contenu des deux types de zones de contexte dans MQMD dépend en fait des options PM\* qui sont spécifiées lors de l'insertion du message. Par conséquent, le contexte d'identité ne se rapporte pas nécessairement à l'application qui a initialement inséré le message, et le contexte d'origine ne se rapporte pas nécessairement à l'application qui a inséré le message le plus récemment-il dépend de la conception de la suite d'applications.

Il existe une classe d'application qui ne modifie jamais le contexte de message, à savoir l'agent MCA (Message Channel Agent). Les agents MCA qui reçoivent des messages des gestionnaires de files d'attente éloignées utilisent l'option de contexte PMSETA dans l'appel MQPUT ou MQPUT1. Cela permet à l'agent MCA récepteur de conserver exactement le contexte de message qui a voyagé avec le message à partir de l'agent MCA émetteur. Toutefois, le résultat est que le contexte d'origine ne se rapporte pas à

l'application qui a le plus récemment inséré le message (l'agent MCA récepteur), mais se rapporte à une application antérieure qui a inséré le message (éventuellement l'application d'origine elle-même).

Pour plus d'informations, voir [Contexte de message](#).

## Date d'expiration du message

Les messages arrivés à expiration dans une file d'attente chargée (une file d'attente qui a été ouverte) sont automatiquement supprimés de la file d'attente dans un délai raisonnable après leur expiration. Certaines autres nouveautés de cette édition de IBM MQ peuvent entraîner une analyse des files d'attente chargées moins fréquente que dans la version précédente du produit, mais les messages arrivés à expiration dans les files d'attente chargées sont toujours supprimés dans un délai raisonnable après leur expiration.

## Zones

La structure MQMD contient les zones suivantes ; les zones sont décrites par ordre alphabétique:






### MDACC (chaîne de 32 octets)

Jeton de comptabilité.

Il fait partie du *contexte d'identité* du message. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

MDACC permet à une application de faire en sorte que le travail effectué à la suite du message soit correctement facturé. Le gestionnaire de files d'attente traite ces informations comme une chaîne de bits et ne vérifie pas leur contenu.

Lorsque le gestionnaire de files d'attente génère ces informations, il est défini comme suit:

- Le premier octet de la zone est défini sur la longueur des informations comptables présentes dans les octets qui suivent ; cette longueur est comprise entre zéro et 30, et est stockée dans le premier octet sous la forme d'un entier binaire.
- Le second octet et les octets suivants (tels que spécifiés par la zone de longueur) sont définis sur les informations de comptabilité appropriées à l'environnement.
  -  **z/OS** Sous z/OS , les informations de comptabilité sont définies sur:
    - Pour le lot z/OS , les informations de comptabilité de la carte JES JOB ou d'une instruction JES ACCT de la carte EXEC (les séparateurs de virgules sont remplacés par X'FF'). Ces informations sont tronquées, si nécessaire, à 31 octets.
    - Pour TSO, numéro de compte de l'utilisateur.
    - Pour CICS, l'identificateur d'unité de travail (UEPUOWDS) de l'unité logique 6.2 (26 octets).
    - Pour IMS, nom de bloc de spécification de programme à 8 caractères concaténé avec le jeton de récupération IMS à 16 caractères.
  -  **IBM i** Sous IBM i, les informations de comptabilité sont définies sur le code de comptabilité du travail.
  -  **Linux**  **AIX** Sous AIX and Linux, les informations de comptabilité sont définies sur l'identificateur numérique de l'utilisateur, en caractères ASCII.
  -  **Windows** Sous Windows, les informations comptables sont définies sur un identificateur de sécurité (SID) Windows NT dans un format compressé. Le SID identifie de manière unique l'identificateur d'utilisateur stocké dans la zone *MDUID* . Lorsque le SID est stocké dans la zone *MDACC* , l'autorité d'identification de 6 octets (située dans le troisième octet et les suivants du SID) est omise. Par exemple, si le SID Windows NT a une longueur de 28 octets, 22 octets d'informations SID sont stockés dans la zone *MDACC* .
- Le dernier octet est défini sur le type de jeton de comptabilité, l'une des valeurs suivantes:

**ATTCIC**

Identificateur CICS LUOW.

**ATTDOS**

Jeton de comptabilité par défaut DOS PC.

**ATTWNT**

Identificateur de sécurité Windows .

**ATT400**

Jeton de comptabilité IBM i .

**ATTUNX**

Identificateur numérique AIX and Linux .

**ATTUSR**

Jeton de comptabilité défini par l'utilisateur.

**ATTENTION**

Type de jeton de comptabilité inconnu.

Le type de jeton de comptabilité est défini sur une valeur explicite uniquement dans les environnements suivants:

-  AIX
-  IBM i
-  Windows

et pour IBM MQ MQI clients connecté à ces systèmes.

Dans d'autres environnements, le type de jeton de comptabilité est défini sur la valeur ATTUNK. Dans ces environnements, la zone MDPAT peut être utilisée pour déduire le type de jeton de comptabilité reçu.

- Tous les autres octets sont définis sur zéro binaire.

Pour les appels MQPUT et MQPUT1 , il s'agit d'une zone d'entrée-sortie si PMSETI ou PMSETA est spécifié dans le paramètre **PMO** . Si ni PMSETI ni PMSETA n'est indiqué, cette zone est ignorée en entrée et est une zone de sortie uniquement. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

Une fois qu'un appel MQPUT ou MQPUT1 a abouti, cette zone contient le MDACC qui a été transmis avec le message s'il a été inséré dans une file d'attente. Il s'agit de la valeur de MDACC qui est conservée avec le message s'il est conservé (voir la description de PMRET dans «MQPMO (options d'insertion de message) sous IBM i», à la page 1230 pour plus de détails sur les publications conservées), mais qui n'est pas utilisée en tant que valeur MDACC lorsque le message est envoyé en tant que publication aux abonnés car ils fournissent une valeur pour remplacer MDACC dans toutes les publications qui leur sont envoyées. Si le message n'a pas de contexte, la zone est entièrement zéro binaire.

Il s'agit d'une zone de sortie pour l'appel MQGET.

Cette zone n'est soumise à aucune conversion basée sur le jeu de caractères du gestionnaire de files d'attente-la zone est traitée comme une chaîne de bits et non comme une chaîne de caractères.

Le gestionnaire de files d'attente ne fait rien avec les informations de cette zone. L'application doit interpréter les informations si elle souhaite les utiliser à des fins comptables.

La valeur spéciale suivante peut être utilisée pour la zone *MDACC* :

**ACNONE**

Aucun jeton de comptabilité n'est spécifié.

La valeur est zéro binaire pour la longueur de la zone.

La longueur de cette zone est indiquée par LNACCT. La valeur initiale de cette zone est ACNONE.

### **MDAID (chaîne de caractères de 32 octets)**

Données d'application relatives à l'identité.

Il fait partie du *contexte d'identité* du message. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

MDAID est une information définie par la suite d'applications et peut être utilisée pour fournir des informations supplémentaires sur le message ou son émetteur. Le gestionnaire de files d'attente traite ces informations comme des données de type caractères, mais ne définit pas leur format. Lorsque le gestionnaire de files d'attente génère ces informations, elles sont entièrement vides.

Pour les appels MQPUT et MQPUT1, il s'agit d'une zone d'entrée-sortie si PMSETI ou PMSETA est spécifié dans le paramètre **PMO**. Si un caractère null est présent, les caractères null et les caractères suivants sont convertis en blancs par le gestionnaire de files d'attente. Si ni PMSETI ni PMSETA n'est indiqué, cette zone est ignorée en entrée et est une zone de sortie uniquement. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

Une fois qu'un appel MQPUT ou MQPUT1 a abouti, cette zone contient le MDAID qui a été transmis avec le message s'il a été inséré dans une file d'attente. Il s'agit de la valeur de MDAID qui est conservée avec le message s'il est conservé (voir la description de PMRET pour plus de détails sur les publications conservées) mais qui n'est pas utilisée en tant que MDAID lorsque le message est envoyé en tant que publication aux abonnés car ils fournissent une valeur pour remplacer MDAID dans toutes les publications qui leur sont envoyées. Si le message n'a pas de contexte, la zone est entièrement vide.

Il s'agit d'une zone de sortie pour l'appel MQGET. La longueur de cette zone est indiquée par LNAIDD. La valeur initiale de cette zone est de 32 caractères blancs.

### **MDAOD (chaîne de caractères de 4 octets)**

Données d'application relatives à l'origine.

Il fait partie du *contexte d'origine* du message. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

MDAOD est une information définie par la suite d'applications qui peut être utilisée pour fournir des informations supplémentaires sur l'origine du message. Par exemple, elle peut être définie par des applications s'exécutant avec des droits utilisateur appropriés pour indiquer si les données d'identité sont dignes de confiance.

Le gestionnaire de files d'attente traite ces informations comme des données de type caractères, mais ne définit pas leur format. Lorsque le gestionnaire de files d'attente génère ces informations, elles sont entièrement vides.

Pour les appels MQPUT et MQPUT1, il s'agit d'une zone d'entrée-sortie si PMSETA est spécifié dans le paramètre **PMO**. Toutes les informations qui suivent un caractère null dans la zone sont supprimées. Le caractère null et tous les caractères suivants sont convertis en blancs par le gestionnaire de files d'attente. Si PMSETA n'est pas indiqué, cette zone est ignorée en entrée et est une zone de sortie uniquement.

Une fois qu'un appel MQPUT ou MQPUT1 a abouti, cette zone contient le MDAOD qui a été transmis avec le message s'il a été inséré dans une file d'attente. Il s'agit de la valeur de MDAOD qui est conservée avec le message s'il est conservé (voir la description de PMRET pour plus de détails sur les publications conservées) mais qui n'est pas utilisée en tant que MDAOD lorsque le message est envoyé en tant que publication aux abonnés car ils fournissent une valeur pour remplacer MDAOD dans toutes les publications qui leur sont envoyées. Si le message n'a pas de contexte, la zone est entièrement vide.

Il s'agit d'une zone de sortie pour l'appel MQGET. La longueur de ce champ est donnée par LNAORD. La valeur initiale de cette zone est de 4 caractères blancs.

### **MDBOC (entier signé à 10 chiffres)**

Compteur d'annulation.

Il s'agit du nombre de fois où le message a été précédemment renvoyé par l'appel MQGET dans le cadre d'une unité d'oeuvre, puis annulé. Il est fourni pour aider l'application à détecter les erreurs de traitement basées sur le contenu des messages. Le nombre exclut les appels MQGET qui ont spécifié l'une des options GMBRW\*.

La précision de ce nombre est affectée par l'attribut de file d'attente **HardenGetBackout** ; voir «Attributs des files d'attente», à la page 1435.

Il s'agit d'une zone de sortie pour l'appel MQGET. Il est ignoré pour les appels MQPUT et MQPUT1 . La valeur initiale de cette zone est 0.

### **MDCID (chaîne de 24 octets)**

Identificateur de corrélation.

Il s'agit d'une chaîne d'octets que l'application peut utiliser pour relier un message à un autre ou pour relier le message à un autre travail effectué par l'application. L'identificateur de corrélation est une propriété permanente du message et est conservé lors des redémarrages du gestionnaire de files d'attente. Etant donné que l'identificateur de corrélation est une chaîne d'octets et non une chaîne de caractères, l'identificateur de corrélation n'est pas converti entre les jeux de caractères lorsque le message est transmis d'un gestionnaire de files d'attente à un autre.

Pour les appels MQPUT et MQPUT1 , l'application peut spécifier n'importe quelle valeur. Le gestionnaire de files d'attente transmet cette valeur avec le message et la distribue à l'application qui émet la demande d'obtention du message.

Si l'application spécifie PMNCID, le gestionnaire de files d'attente génère un identificateur de corrélation unique qui est envoyé avec le message et renvoyé à l'application émettrice lors de la sortie de l'appel MQPUT ou MQPUT1 .

Cet identificateur de corrélation généré est conservé avec le message s'il est conservé et est utilisé comme identificateur de corrélation lorsque le message est envoyé en tant que publication aux abonnés qui spécifient CINONE dans la zone SDCID du MQSD transmis sur l'appel MQSUB.

Voir «MQPMO (options d'insertion de message) sous IBM i», à la page 1230 pour plus de détails sur les publications conservées

Lorsque le gestionnaire de files d'attente ou un agent MCA génère un message de rapport, il définit la zone MDCID de la manière indiquée par la zone MDREP du message d'origine, ROCMTC ou ROPCI. Les applications qui génèrent des messages de rapport doivent également le faire.

Pour l'appel MQGET, MDCID est l'une des cinq zones qui peuvent être utilisées pour sélectionner un message particulier à extraire de la file d'attente. Voir la description de la zone MDMID pour plus de détails sur la manière de spécifier des valeurs pour cette zone.

Le fait de spécifier CINONE comme identificateur de corrélation a le même effet que le fait de ne pas spécifier MOCORI, c'est-à-dire que tout identificateur de corrélation correspond.

Si l'option GMMUC est spécifiée dans le paramètre **GMO** de l'appel MQGET, cette zone est ignorée.

En cas de retour d'un appel MQGET, la zone MDCID est définie sur l'identificateur de corrélation du message renvoyé (le cas échéant).

Les valeurs spéciales suivantes peuvent être utilisées:

#### **CINONE**

Aucun identificateur de corrélation n'est indiqué.

La valeur est zéro binaire pour la longueur de la zone.

#### **CINEWS**

Le message est le début d'une nouvelle session.

Cette valeur est reconnue par CICS bridge comme indiquant le début d'une nouvelle session, c'est-à-dire le début d'une nouvelle séquence de messages.

Pour l'appel MQGET, il s'agit d'une zone d'entrée/sortie. Pour les appels MQPUT et MQPUT1, il s'agit d'une zone d'entrée si PMNCID n'est pas spécifié et d'une zone de sortie si PMNCID est spécifié. La longueur de cette zone est indiquée par LNCID. La valeur initiale de cette zone est CINONE.

### **MDCSI (entier signé à 10 chiffres)**

Indique l'identificateur de jeu de caractères des données de type caractère dans le message.

**Remarque :** Les données de type caractère dans MQMD et les autres structures de données IBM MQ qui sont des paramètres des appels doivent être dans le jeu de caractères du gestionnaire de files d'attente. Il est défini par l'attribut **CodedCharSetId** du gestionnaire de files d'attente ; pour plus de détails sur cet attribut, voir «Attributs du gestionnaire de files d'attente sous IBM i», à la page 1469.

Les valeurs spéciales suivantes peuvent être utilisées:

#### **CQM**

Identificateur de jeu de caractères du gestionnaire de files d'attente.

Les données de type caractère du message se trouvent dans le jeu de caractères du gestionnaire de files d'attente.

Dans les appels MQPUT et MQPUT1, le gestionnaire de files d'attente remplace cette valeur dans le MQMD envoyé avec le message par l'identificateur de jeu de caractères true du gestionnaire de files d'attente. Par conséquent, la valeur CSQM n'est jamais renvoyée par l'appel MQGET.

#### **CSINHT**

Hérite de l'identificateur de jeu de caractères de cette structure.

Les données de type caractère du message se trouvent dans le même jeu de caractères que cette structure ; il s'agit du jeu de caractères du gestionnaire de files d'attente. (Pour MQMD uniquement, CSINHT a la même signification que CSQM).

Le gestionnaire de files d'attente modifie cette valeur dans le MQMD envoyé avec le message en indiquant l'identificateur de jeu de caractères réel de MQMD. Si aucune erreur ne se produit, la valeur CSINHT n'est pas renvoyée par l'appel MQGET.

CSINHT ne peut pas être utilisé si la valeur de la zone MDPAT dans MQMD est ATBRKR.

#### **CSEMBD**

Identificateur de jeu de caractères imbriqué.

Les données de type caractère du message se trouvent dans un jeu de caractères avec l'identificateur contenu dans les données du message lui-même. Il peut y avoir n'importe quel nombre d'identificateurs de jeu de caractères imbriqués dans les données de message, s'appliquant à différentes parties des données. Cette valeur doit être utilisée pour les messages PCF contenant des données dans une combinaison de jeux de caractères. Les messages PCF ont un nom de format FMPCF.

Indiquez cette valeur uniquement sur les appels MQPUT et MQPUT1. S'il est spécifié dans l'appel MQGET, il empêche la conversion du message.

Dans les appels MQPUT et MQPUT1, le gestionnaire de files d'attente modifie les valeurs CSQM et CSINHT dans le MQMD envoyé avec le message comme décrit précédemment, mais ne modifie pas le MQMD spécifié dans l'appel MQPUT ou MQPUT1. Aucune autre vérification n'est effectuée sur la valeur spécifiée.

Les applications qui extraient des messages doivent comparer cette zone à la valeur attendue par l'application. Si les valeurs diffèrent, l'application peut avoir besoin de convertir des données de type caractères dans le message.

Si l'option GMCONV est spécifiée dans l'appel MQGET, cette zone est une zone d'entrée-sortie. La valeur indiquée par l'application est l'identificateur de jeu de caractères codés dans lequel les données du message doivent être converties si nécessaire. Si la conversion aboutit ou est inutile, la valeur reste inchangée (sauf que la valeur CSQM ou CSINHT est convertie en valeur réelle). Si la conversion échoue, la valeur après l'appel MQGET représente l'identificateur de jeu de caractères codés du message non converti renvoyé à l'application.



Sinon, il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La valeur initiale de cette zone est CSQM.

### **MDENC (entier signé à 10 chiffres)**

Codage numérique des données de message.

Indique le codage numérique des données numériques dans le message ; il ne s'applique pas aux données numériques dans la structure MQMD elle-même. Le codage numérique définit la représentation utilisée pour les entiers binaires, les entiers décimaux condensés et les nombres à virgule flottante.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Le gestionnaire de files d'attente ne vérifie pas que la zone est valide. La valeur spéciale suivante est définie:

#### **ENNAT**

Codage de la machine native.

Le codage est la valeur par défaut du langage de programmation et de la machine sur lesquels l'application s'exécute.

**Remarque :** La valeur de cette constante dépend du langage de programmation et de l'environnement. Pour cette raison, les applications doivent être compilées à l'aide des fichiers d'en-tête, de macro, de COPY ou d'INCLUDE appropriés à l'environnement dans lequel l'application sera exécutée.

Les applications qui placent des messages doivent normalement spécifier ENNAT. Les applications qui extraient des messages doivent comparer cette zone à la valeur ENNAT ; si les valeurs diffèrent, l'application peut avoir besoin de convertir des données numériques dans le message. L'option GMCONV peut être utilisée pour demander au gestionnaire de files d'attente de convertir le message dans le cadre du traitement de l'appel MQGET.

Si l'option GMCONV est spécifiée dans l'appel MQGET, cette zone est une zone d'entrée-sortie. La valeur spécifiée par l'application est le codage dans lequel les données de message doivent être converties si nécessaire. Si la conversion aboutit ou n'est pas nécessaire, la valeur reste inchangée. Si la conversion échoue, la valeur après l'appel MQGET représente le codage du message non converti renvoyé à l'application.

Dans d'autres cas, il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La valeur initiale de cette zone est ENNAT.

### **MDEXP (entier signé à 10 chiffres)**

Durée de vie du message.

Il s'agit d'une période de temps exprimée en dixièmes de seconde, définie par l'application qui insère le message. Le message peut être supprimé s'il n'a pas été supprimé de la file d'attente de destination avant l'expiration de ce délai.

La valeur est décrétementée pour refléter le temps passé par le message dans la file d'attente de destination, ainsi que dans les files d'attente de transmission intermédiaires si l'insertion est dans une file d'attente éloignée. Elle peut également être décrétementée par des agents de canal de message pour refléter les temps de transmission, si ceux-ci sont significatifs. De même, une application qui transmet ce message à une autre file d'attente peut décrétement la valeur si nécessaire, si elle a conservé le message pendant un temps significatif. Toutefois, le délai d'expiration est traité comme approximatif et la valeur n'a pas besoin d'être décrétementée pour refléter de petits intervalles de temps.

Lorsque le message est extrait par une application à l'aide de l'appel MQGET, la zone MDEXP représente la durée d'expiration d'origine qui reste.

Une fois le délai d'expiration d'un message écoulé, il peut être supprimé par le gestionnaire de files d'attente. Dans les implémentations en cours, le message est supprimé lorsqu'un appel MQGET de recherche ou de non-recherche est émis qui aurait renvoyé le message s'il n'avait pas déjà expiré. Par exemple, un appel MQGET de non-consultation avec la zone GMMO de MQGMO définie

sur MONONE en lecture à partir d'une file d'attente ordonnée FIFO entraîne la suppression de tous les messages arrivés à expiration jusqu'au premier message non arrivé à expiration. Avec une file d'attente ordonnée de priorité, le même appel va supprimer les messages arrivés à expiration de priorité supérieure et les messages de priorité égale qui sont arrivés dans la file d'attente avant le premier message non arrivé à expiration.

Un message arrivé à expiration n'est jamais renvoyé à une application (par un appel MQGET de recherche ou de non-recherche), de sorte que la valeur de la zone MDEXP du descripteur de message après un appel MQGET réussi est supérieure à zéro ou la valeur spéciale EIULIM.

Si un message est inséré dans une file d'attente éloignée, il peut expirer (et être supprimé) alors qu'il se trouve dans une file d'attente de transmission intermédiaire, avant que le message n'atteigne la file d'attente de destination.

Un rapport est généré lorsqu'un message arrivé à expiration est supprimé, si le message a spécifié l'une des options de rapport ROEXP\*. Si aucune de ces options n'est spécifiée, aucun rapport de ce type n'est généré ; le message est supposé ne plus être pertinent après cette période (peut-être parce qu'un message ultérieur l'a remplacé).

Tout autre programme qui supprime des messages en fonction de l'heure d'expiration doit également envoyer un message de rapport approprié si un tel message a été demandé.

**Remarque :**

1. Si un message est inséré avec une heure MDEXP égale à zéro, l'appel MQPUT ou MQPUT1 échoue avec le code anomalie RC2013; aucun message de rapport n'est généré dans ce cas.
2. Etant donné qu'un message dont le délai d'expiration est écoulé peut ne pas être supprimé avant une date ultérieure, il se peut que des messages d'une file d'attente aient dépassé leur délai d'expiration et ne soient donc pas éligibles à l'extraction. Ces messages sont néanmoins pris en compte dans le nombre de messages de la file d'attente à toutes fins, y compris le déclenchement de la profondeur.
3. Un rapport d'expiration est généré, s'il est demandé, lorsque le message est effectivement supprimé, et non lorsqu'il devient éligible à la suppression.
4. La suppression d'un message arrivé à expiration et la génération d'un rapport d'expiration, le cas échéant, ne font jamais partie de l'unité de travail de l'application, même si la suppression du message a été planifiée suite à un appel MQGET opérant dans une unité de travail.
5. Si un message presque arrivé à expiration est extrait par un appel MQGET au sein d'une unité de travail et que l'unité de travail est ensuite annulée, le message peut être supprimé avant de pouvoir être à nouveau extrait.
6. Si un message presque arrivé à expiration est verrouillé par un appel MQGET avec GMLK, le message peut devenir éligible pour être supprimé avant de pouvoir être extrait par un appel MQGET avec GMMUC ; le code anomalie RC2034 est renvoyé sur cet appel MQGET suivant si cela se produit.
7. Lorsqu'un message de demande dont le délai d'expiration est supérieur à zéro est extrait, l'application peut effectuer l'une des actions suivantes lorsqu'elle envoie le message de réponse:
  - Copiez le délai d'expiration restant entre le message de demande et le message de réponse.
  - Définissez l'heure d'expiration dans le message de réponse sur une valeur explicite supérieure à zéro.
  - Définissez l'heure d'expiration dans le message de réponse à EIULIM.

L'action à effectuer dépend de la conception de la suite d'applications. Toutefois, l'action par défaut pour placer des messages dans une file d'attente de messages non livrés doit être de conserver le délai d'expiration restant du message et de continuer à le décrémenter.

8. Les messages de déclenchement sont toujours générés avec EIULIM.
9. Un message (normalement dans une file d'attente de transmission) dont le nom MDFMT est FMXQH possède un deuxième descripteur de message dans le MQXQH. Il est donc associé à deux zones MDEXP . Dans ce cas, il convient de noter les points supplémentaires suivants:

- Lorsqu'une application place un message dans une file d'attente éloignée, le gestionnaire de files d'attente place initialement le message dans une file d'attente de transmission locale et préfixe les données de message d'application avec une structure MQXQH. Le gestionnaire de files d'attente définit les valeurs des deux zones MDEXP de sorte qu'elles soient identiques à celles spécifiées par l'application.

Si une application insère un message directement dans une file d'attente de transmission locale, les données de message doivent déjà commencer par une structure MQXQH et le nom de format doit être FMXQH (mais le gestionnaire de files d'attente ne l'applique pas). Dans ce cas, l'application n'a pas besoin de définir les valeurs de ces deux zones MDEXP pour qu'elles soient identiques. (Le gestionnaire de files d'attente ne vérifie pas que la zone MDEXP dans le MQXQH contient une valeur valide, ni même que les données de message sont suffisamment longues pour l'inclure.)

- Lorsqu'un message portant le nom MDFMT FMXQH est extrait d'une file d'attente (qu'il s'agisse d'une file d'attente normale ou d'une file d'attente de transmission), le gestionnaire de files d'attente décrémente ces deux zones MDEXP avec le temps d'attente de la file d'attente. Aucune erreur n'est générée si les données de message ne sont pas suffisamment longues pour inclure la zone MDEXP dans le MQXQH.
- Le gestionnaire de files d'attente utilise la zone MDEXP dans le descripteur de message distinct (c'est-à-dire, pas celui du descripteur de message intégré dans la structure MQXQH) pour tester si le message peut être supprimé.
- Si les valeurs initiales des deux zones MDEXP étaient différentes, il est donc possible que l'heure MDEXP dans le descripteur de message distinct lorsque le message est extrait soit supérieure à zéro (de sorte que le message ne peut pas être supprimé), alors que l'heure indiquée dans la zone MDEXP de MQXQH est écoulée. Dans ce cas, la zone MDEXP de MQXQH est définie sur zéro.

La valeur spéciale suivante est reconnue:

#### **EIULIM**

Durée de vie illimitée.

Le message a un délai d'expiration illimité.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1. La valeur initiale de cette zone est EIULIM.

#### **MDFB (entier signé à 10 chiffres)**

Retour d'informations ou code anomalie.

Elle est utilisée avec un message de type MTRPRT pour indiquer la nature du rapport et n'est significative qu'avec ce type de message. La zone peut contenir l'une des valeurs FB\* ou l'une des valeurs RC\*. Les codes retour sont regroupés comme suit:

#### **FBNONE**

Aucun commentaire n'a été fourni.

#### **FBSFST**

Valeur la plus faible pour les commentaires générés par le système.

#### **FBSLST**

Valeur la plus élevée pour les commentaires générés par le système.

La plage des codes retour générés par le système FBSFST à FBSLST inclut les codes retour généraux répertoriés plus loin dans cette section (FB\*), ainsi que les codes raison (RC\*) qui peuvent se produire lorsque le message ne peut pas être placé dans la file d'attente de destination.

#### **FBAFST**

Valeur la plus faible pour les commentaires générés par l'application.

#### **FBALST**

Valeur la plus élevée pour les commentaires générés par l'application.

Les applications qui génèrent des messages de rapport ne doivent pas utiliser de codes retour dans la plage système (autre que FBQUIT), sauf si elles souhaitent simuler des messages de rapport générés par le gestionnaire de files d'attente ou l'agent MCA.

Dans les appels MQPUT ou MQPUT1, la valeur indiquée doit être FBNONE ou être comprise dans la plage système ou la plage d'applications. Cette case est cochée quelle que soit la valeur de MDMT.

### **Codes retour généraux:**

#### **FBCOA**

Confirmation de l'arrivée dans la file d'attente de destination (voir ROCOA).

#### **FBCOD**

Confirmation de la livraison à la demande de réception (voir ROCOD).

#### **FBEXP**

Le message est arrivé à expiration.

Le message a été supprimé car il n'avait pas été supprimé de la file d'attente de destination avant que son délai d'expiration ne soit écoulé.

#### **FBPAN**

Notification d'action positive (voir ROPAN).

#### **FBNAN**

Notification d'action négative (voir RONAN).

#### **FBQUIT**

L'application doit se terminer.

Cela peut être utilisé par un programme de planification de la charge de travail pour contrôler le nombre d'instances d'un programme d'application en cours d'exécution. L'envoi d'un message MTRPRT avec ce code retour à une instance du programme d'application indique à cette instance qu'elle doit arrêter le traitement. Toutefois, le respect de cette convention est du ressort de l'application ; elle n'est pas appliquée par le gestionnaire de files d'attente.

**IMS-bridge feedback codes:** lorsque le pont IMS reçoit un code de détection IMS-OTMA différent de zéro, le pont IMS convertit le code de détection de la valeur hexadécimale en valeur décimale, ajoute la valeur FBIERR (300) et place le résultat dans la zone MDFB du message de réponse. Il en résulte que le code retour a une valeur comprise entre FBIFST (301) et FBILST (399) lorsqu'une erreur IMS-OTMA s'est produite.

Les codes retour suivants peuvent être générés par le pont IMS :

#### **FBDLZ**

Longueur des données égale à zéro.

La longueur d'un segment était égale à zéro dans les données d'application du message.

#### **FBDLN**

Longueur de données négative.

Une longueur de segment était négative dans les données d'application du message.

#### **FBDLTB**

Longueur de données trop grande.

Une longueur de segment était trop grande dans les données d'application du message.

#### **FBBUFO**

Dépassement de la mémoire tampon.

La valeur de l'une des zones de longueur entraînerait le dépassement de la mémoire tampon de messages par les données.

#### **FBLOB1**

Longueur en erreur de 1.

La valeur de l'une des zones de longueur était un octet trop court.

**FBIIH**

Structure MQIIH non valide ou manquante.

La zone MDFMT dans MQMD indique FMIMS, mais le message ne commence pas par une structure MQIIH valide.

**FBANI**

ID utilisateur non autorisé à être utilisé dans IMS.

L'ID utilisateur contenu dans le descripteur de message MQMD ou le mot de passe contenu dans la zone IIAUT de la structure MQIIH n'a pas pu être validé par le pont IMS . Par conséquent, le message n'a pas été transmis à IMS.

**FBIERR**

Erreur inattendue renvoyée par IMS.

Une erreur inattendue a été renvoyée par IMS. Consultez le journal des erreurs IBM MQ sur le système sur lequel réside le pont IMS pour plus d'informations sur l'erreur.

**FBIFST**

Valeur la plus basse pour les commentaires générés par IMS.

Les codes retour générés par IMS occupent la plage FBIFST (300) à FBILST (399). Le code de détection IMS-OTMA lui-même est MDFB moins FBIERR.

**FBILST**

Valeur la plus élevée pour les commentaires générés par IMS.

**CICS-bridge feedback codes:** les codes retour suivants peuvent être générés par CICS bridge:

**FBCAAB**

Arrêt anormal de l'application.

Le programme d'application indiqué dans le message s'est arrêté de manière anormale. Ce code retour apparaît uniquement dans la zone DLREA de la structure MQDLH.

**FBCANS**

L'application ne peut pas être démarrée.

La commande EXEC CICS LINK pour le programme d'application spécifié dans le message a échoué. Ce code retour apparaît uniquement dans la zone DLREA de la structure MQDLH.

**FBCBRF**

CICS bridge s'est arrêté de manière anormale sans avoir terminé le traitement d'erreur normal.

**FBCCSSE**

Identificateur de jeu de caractères incorrect.

**FBCIHE**

Structure d'en-tête d'informations CICS manquante ou non valide.

**FBCCAE**

La longueur de la zone de communication CICS n'est pas valide.

**FBCCEI**

Identificateur de corrélation incorrect.

**FBCDLQ**

File d'attente de rebut non disponible.

La tâche CICS bridge n'a pas pu copier une réponse à cette demande dans la file d'attente de rebut. La demande a été annulée.

**FBCENE**

Codage non valide.

**FBCINE**

CICS bridge a rencontré une erreur inattendue.

Ce code retour apparaît uniquement dans la zone DLREA de la structure MQDLH.

**FBCNTA**

ID utilisateur non autorisé ou mot de passe incorrect.

Ce code retour apparaît uniquement dans la zone DLREA de la structure MQDLH.

**Objet FBCUBO**

Unité de travail annulée.

L'unité de travail a été annulée pour l'une des raisons suivantes:

- Une erreur a été détectée lors du traitement d'une autre demande dans la même unité d'oeuvre.
- Une fin anormale CICS s'est produite alors que l'unité de travail était en cours.

**FBCUWE**

Zone de contrôle d'unité de travail CIUOW incorrecte.

**Codes anomalieMQ:** pour les messages de rapport d'exception, MDFB contient un code anomalie MQ . Les codes anomalie possibles sont les suivants:

**RC2051**

(2051, X'803') Appels d'insertion bloqués pour la file d'attente.

**RC2053**

(2053, X'805') La file d'attente contient déjà le nombre maximal de messages.

**RC2035**

(2035, X'7F3') Non autorisé pour l'accès.

**RC2056**

(2056, X'808') Aucun espace disponible sur le disque pour la file d'attente.

**RC2048**

(2048, X'800') La file d'attente ne prend pas en charge les messages persistants.

**RC2031**

(2031, X'7EF') Longueur de message supérieure à la longueur maximale pour le gestionnaire de files d'attente.

**RC2030**

(2030, X'7EE') Longueur de message supérieure au maximum pour la file d'attente.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La valeur initiale de cette zone est FBNONE.

**MDFMT (chaîne de caractères de 8 octets)**

Nom de format des données de message.

Il s'agit d'un nom que l'expéditeur du message peut utiliser pour indiquer au destinataire la nature des données du message. Tous les caractères du jeu de caractères du gestionnaire de files d'attente peuvent être spécifiés pour le nom, mais il est recommandé de limiter le nom aux éléments suivants:

- Majuscules de A à Z
- Chiffres de 0 à 9

Si d'autres caractères sont utilisés, il est possible qu'il ne soit pas possible de convertir le nom entre les jeux de caractères des gestionnaires de files d'attente d'envoi et de réception.

Le nom doit être rempli avec des blancs à la longueur de la zone, ou un caractère indéfini utilisé pour terminer le nom avant la fin de la zone ; la valeur indéfinie et les caractères suivants sont traités comme des blancs. N'indiquez pas de nom avec des espaces de début ou des blancs imbriqués. Pour l'appel MQGET, le gestionnaire de files d'attente renvoie le nom complété par des blancs à la longueur de la zone.

Le gestionnaire de files d'attente ne vérifie pas que le nom est conforme aux recommandations décrites précédemment.

Les noms commençant par "MQ" en majuscules, en minuscules et en casse mixte ont des significations qui sont définies par le gestionnaire de files d'attente ; vous ne devez pas utiliser de

noms commençant par ces lettres pour vos propres formats. Les formats intégrés du gestionnaire de files d'attente sont les suivants:

#### **FMAUCUN**

Aucun nom de format.

La nature des données n'est pas définie. Cela signifie que les données ne peuvent pas être converties lorsque le message est extrait d'une file d'attente à l'aide de l'option GMCONV.

Si GMCONV est spécifié dans l'appel MQGET et que le jeu de caractères ou le codage des données du message diffère de celui spécifié dans le paramètre **MSGDSC**, le message est renvoyé avec les codes achèvement et anomalie suivants (en supposant qu'il n'y ait pas d'autres erreurs):

- Code achèvement CCWARN et code anomalie RC2110 si les données FMNONE sont au début du message.
- Code achèvement CCOK et code raison RCNONE si les données FMNONE sont à la fin du message (c'est-à-dire précédées d'une ou de plusieurs structures d'en-tête MQ). Les structures d'en-tête MQ sont converties au jeu de caractères et au codage demandés dans ce cas.

#### **FMADMN**

Message de demande / réponse du serveur de commandes.

Le message est un message de demande ou de réponse du serveur de commandes au format PCF (Programmable Command Format). Les messages de ce format peuvent être convertis si l'option GMCONV est spécifiée dans l'appel MQGET. Pour plus d'informations sur l'utilisation des messages au format de commande programmable, voir [Utilisation des formats de commande programmables](#).

#### **modulation de fréquenceCICS**

En-tête d'informations CICS .

Les données de message commencent par l'en-tête d'information CICS MQCIH, qui est suivi des données d'application. Le nom de format des données d'application est donné par la zone CIFMT dans la structure MQCIH.

#### **FMCMD1**

Message de réponse à la commande de type 1.

Le message est un message de réponse du serveur de commandes MQSC contenant le nombre d'objets, le code achèvement et le code anomalie. Les messages de ce format peuvent être convertis si l'option GMCONV est spécifiée dans l'appel MQGET.

#### **FMCMD2**

Message de réponse à la commande de type 2.

Le message est un message de réponse du serveur de commandes MQSC contenant des informations sur le ou les objets demandés. Les messages de ce format peuvent être convertis si l'option GMCONV est spécifiée dans l'appel MQGET.

#### **FMDLH**

En-tête de rebut.

Les données de message commencent par l'en-tête de rebut MQDLH. Les données du message d'origine suivent immédiatement la structure MQDLH. Le nom de format des données de message d'origine est donné par la zone DLFMT dans la structure MQDLH ; pour plus de détails sur cette structure, voir [«MQDLH \(en-tête de rebut\) sur IBM i»](#), à la page 1115 . Les messages de ce format peuvent être convertis si l'option GMCONV est spécifiée dans l'appel MQGET.

Les rapports COA et COD ne sont pas générés pour les messages dont le MDFMT est FMDLH.

#### **FMDH**

En-tête de liste de distribution.

Les données de message commencent par l'en-tête de liste de distribution MQDH ; cela inclut les tableaux des enregistrements MQOR et MQPMR. L'en-tête de la liste de distribution peut être suivi de données supplémentaires. Le format des données supplémentaires (le cas échéant) est fourni

par la zone DHFMT dans la structure MQDH ; voir «MQDH (en-tête de distribution) sur IBM i», à la [page 1110](#) pour plus de détails sur cette structure. Les messages au format FMDH peuvent être convertis si l'option GMCONV est spécifiée dans l'appel MQGET.

### **FMEVNT**

Message d'événement.

Le message est un message d'événement MQ qui signale un événement qui s'est produit. Les messages d'événement ont la même structure que les commandes programmables ; pour plus d'informations sur cette structure, voir [Structures des commandes et des réponses](#). Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

Les messages d'événement Version-1 peuvent être convertis si l'option GMCONV est spécifiée dans l'appel MQGET.

### **modulation de fréquenceIMS**

En-tête d'informations IMS .

Les données de message commencent par l'en-tête d'information IMS MQIIH, qui est suivi des données d'application. Le nom de format des données d'application est donné par la zone *IIFMT* dans la structure MQIIH. Les messages de ce format peuvent être convertis si l'option GMCONV est spécifiée dans l'appel MQGET.

### **FMIMVS**

Chaîne de variable IMS .

Le message est une chaîne de variable IMS , qui est une chaîne au format 11zzccc, où:

#### **11**

est une zone de longueur de 2 octets spécifiant la longueur totale de l'élément de chaîne de variable IMS . Cette longueur est égale à la longueur de 11 (2 octets), plus la longueur de zz (2 octets), plus la longueur de la chaîne de caractères elle-même. 11 est un entier binaire à 2 octets dans le codage spécifié par la zone MDENC .

#### **zz**

est une zone à 2 octets contenant des indicateurs significatifs pour IMS. zz est une chaîne d'octets composée de deux zones de chaîne de bits à 1 octet et est transmise sans modification de l'expéditeur au récepteur (c'est-à-dire que zz n'est soumis à aucune conversion).

#### **ccc**

est une chaîne de caractères de longueur variable contenant 11 -4 caractères. ccc est dans le jeu de caractères spécifié par la zone MDCSI .

Les messages de ce format peuvent être convertis si l'option GMCONV est spécifiée dans l'appel MQGET.

### **FMMDE**

Extension du descripteur de message.

Les données de message commencent par l'extension de descripteur de message MQMDE et sont éventuellement suivies par d'autres données (généralement les données de message d'application). Le nom de format, le jeu de caractères et le codage des données qui suivent MQMDE sont fournis par les zones MEFMT, MECSI et MEENC dans MQMDE. Pour plus de détails sur cette structure, voir «MQMDE (extension de descripteur de message) sous IBM i», à la [page 1209](#) . Les messages de ce format peuvent être convertis si l'option GMCONV est spécifiée dans l'appel MQGET.

### **FMPCF**

Message défini par l'utilisateur au format PCF (Programmable Command Format).

Le message est un message défini par l'utilisateur qui est conforme à la structure d'un message PCF (Programmable Command Format). Les messages de ce format peuvent être convertis si l'option GMCONV est spécifiée dans l'appel MQGET. Pour plus d'informations sur l'utilisation des



messages au format de commande programmable, voir [Utilisation des formats de commande programmables](#) .

#### **FMRMH**

En-tête de message de référence.

Les données de message commencent par l'en-tête de message de référence MQRMH et sont éventuellement suivies d'autres données. Le nom de format, le jeu de caractères et le codage des données sont donnés par les zones RMFMT, RMCSIet RMENC dans MQRMH. Pour plus de détails sur cette structure, voir «[MQRMH \(en-tête de message de référence\) sur IBM i](#)», à la page 1258 . Les messages de ce format peuvent être convertis si l'option GMCONV est spécifiée dans l'appel MQGET.

#### **FMRFH**

Règles et en-tête de formatage.

Les données de message commencent par les règles et l'en-tête de formatage MQRFH, et sont éventuellement suivies par d'autres données. Le nom de format, le jeu de caractères et le codage des données (le cas échéant) sont fournis par les zones RFFMT, RFCSIet RFENC dans le MQRFH. Les messages de ce format peuvent être convertis si l'option GMCONV est spécifiée dans l'appel MQGET.

#### **FMRFH2**

Règles et formatage de l'en-tête version 2.

Les données de message commencent par les règles version-2 et l'en-tête de formatage MQRFH2, et sont éventuellement suivies par d'autres données. Le nom de format, le jeu de caractères et le codage des données facultatives (le cas échéant) sont fournis par les zones RF2FMT, RF2CSIet RF2ENC dans MQRFH2. Les messages de ce format peuvent être convertis si l'option GMCONV est spécifiée dans l'appel MQGET.

#### **FSTR**

Message composé entièrement de caractères.

Les données de message d'application peuvent être soit une chaîne SBCS (jeu de caractères codé sur un octet), soit une chaîne DBCS (jeu de caractères codé sur deux octets). Les messages de ce format peuvent être convertis si l'option GMCONV est spécifiée dans l'appel MQGET.

#### **FMTM**

Message de déclenchement.

Le message est un message de déclenchement, décrit par la structure MQTM. Pour plus de détails sur cette structure, voir «[MQTM-Message de déclenchement](#)», à la page 1296 . Les messages de ce format peuvent être convertis si l'option GMCONV est spécifiée dans l'appel MQGET.

#### **FMWIH**

En-tête d'informations de travail.

Les données de message commencent par l'en-tête d'informations de travail MQWIH, qui est suivi des données d'application. Le nom de format des données d'application est donné par la zone WIFMT dans la structure MQWIH.

#### **FMXQH**

En-tête de file d'attente de transmission.

Les données de message commencent par l'en-tête de file d'attente de transmission MQXQH. Les données du message d'origine suivent immédiatement la structure MQXQH. Le nom de format des données de message d'origine est fourni par la zone MDFMT de la structure MQMD qui fait partie de l'en-tête de file d'attente de transmission MQXQH. Pour plus de détails sur cette structure, voir «[MQXQH \(en-tête de file d'attente de transmission\) sous IBM i](#)», à la page 1306 .

Les rapports COA et COD ne sont pas générés pour les messages dont le MDFMT est FMXQH.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La longueur de cette zone est indiquée par LNFMT. La valeur initiale de cette zone est FMNONE.

## MDGID (chaîne de 24 octets)

Identificateur de groupe.

Il s'agit d'une chaîne d'octets utilisée pour identifier le groupe de messages particulier ou le message logique auquel appartient le message physique. MDGID est également utilisé si la segmentation est autorisée pour le message. Dans tous ces cas, MDGID a une valeur non nulle et un ou plusieurs des indicateurs suivants sont définis dans la zone MDMFL :

- MFMIG
- MFLMIG
- MFSEG
- MFLSEG
- MFSEGA

Si aucun de ces indicateurs n'est défini, MDGID a la valeur null spéciale GINONE.

Cette zone n'a pas besoin d'être définie par l'application sur l'appel MQPUT ou MQGET si:

- Dans l'appel MQPUT, PMLOGO est spécifié.
- Dans l'appel MQGET, MOGRPI n'est pas spécifié.

Envisagez d'utiliser ces appels pour les messages qui ne sont pas des messages de rapport. Toutefois, si l'application requiert davantage de contrôle ou si l'appel est MQPUT1, l'application doit s'assurer que MDGID est défini sur une valeur appropriée.

Les groupes de messages et les segments ne peuvent être traités correctement que si l'identificateur de groupe est unique. Pour cette raison, les applications ne doivent pas générer leurs propres identificateurs de groupe ; à la place, les applications doivent effectuer l'une des opérations suivantes:

- Si PMLOGO est spécifié, le gestionnaire de files d'attente génère automatiquement un identificateur de groupe unique pour le premier message du groupe ou segment du message logique et utilise cet identificateur de groupe pour les messages restants du groupe ou des segments du message logique, de sorte que l'application n'a pas besoin d'effectuer d'action spéciale. Envisagez d'utiliser cette procédure.
- Si PMLOGO n'est pas spécifié, l'application doit demander au gestionnaire de files d'attente de générer l'identificateur de groupe, en définissant MDGID sur GINONE lors du premier appel MQPUT ou MQPUT1 pour un message dans le groupe ou le segment du message logique. L'identificateur de groupe renvoyé par le gestionnaire de files d'attente en sortie de cet appel doit ensuite être utilisé pour les messages restants du groupe ou des segments du message logique. Si un groupe de messages contient des messages segmentés, le même identificateur de groupe doit être utilisé pour tous les segments et messages du groupe.

Lorsque PMLOGO n'est pas spécifié, les messages des groupes et les segments de messages logiques peuvent être placés dans n'importe quel ordre (par exemple, dans l'ordre inverse), mais l'identificateur de groupe doit être alloué par le premier appel MQPUT ou MQPUT1 émis pour l'un de ces messages.

Lors de l'entrée dans les appels MQPUT et MQPUT1, le gestionnaire de files d'attente utilise la valeur détaillée dans [PMOPT](#). Dans la sortie des appels MQPUT et MQPUT1, le gestionnaire de files d'attente définit cette zone sur la valeur qui a été envoyée avec le message si l'objet ouvert est une file d'attente unique et non une liste de distribution, mais la laisse inchangée si l'objet ouvert est une liste de distribution. Dans ce dernier cas, si l'application doit connaître les identificateurs de groupe générés, elle doit fournir des enregistrements MQPMR contenant la zone PRGID.

En entrée de l'appel MQGET, le gestionnaire de files d'attente utilise la valeur détaillée dans le [Tableau 1](#). Dans la sortie de l'appel MQGET, le gestionnaire de files d'attente définit cette zone sur la valeur du message extrait.

La valeur spéciale suivante est définie:

## GINONE

Aucun identificateur de groupe n'a été indiqué.

La valeur est zéro binaire pour la longueur de la zone. Il s'agit de la valeur utilisée pour les messages qui ne sont pas dans des groupes, qui ne sont pas des segments de messages logiques et pour lesquels la segmentation n'est pas autorisée.

La longueur de cette zone est indiquée par LNGID. La valeur initiale de cette zone est GINONE. Cette zone est ignorée si MDVER est inférieur à MDVER2.

## MDMFL (entier signé à 10 chiffres)

Indicateurs de message.

Il s'agit d'indicateurs qui spécifient les attributs du message ou qui contrôlent son traitement. Les indicateurs sont répartis dans les catégories suivantes:

- Indicateur de segmentation
- Indicateurs de statut

Ceux-ci sont décrits à leur tour.

**Indicateurs de segmentation:** Lorsqu'un message est trop volumineux pour une file d'attente, une tentative d'insertion du message dans la file d'attente échoue généralement. La segmentation est une technique par laquelle le gestionnaire de files d'attente ou l'application fractionne le message en éléments plus petits appelés segments et place chaque segment dans la file d'attente sous la forme d'un message physique distinct. L'application qui extrait le message peut soit extraire les segments un par un, soit demander au gestionnaire de files d'attente de réassembler les segments en un seul message renvoyé par l'appel MQGET. Ce dernier est obtenu en spécifiant l'option GMCMPM sur l'appel MQGET et en fournissant une mémoire tampon suffisamment grande pour contenir le message complet. (Pour plus de détails sur l'option GMCMPM, voir «MQGMO (options d'extraction de message) sous IBM i», à la page 1127.) La segmentation d'un message peut se produire sur le gestionnaire de files d'attente émetteur, sur un gestionnaire de files d'attente intermédiaire ou sur le gestionnaire de files d'attente de destination.

Vous pouvez spécifier l'une des options suivantes pour contrôler la segmentation d'un message:

### MFSEGI

Segmentation inhibée.

Cette option empêche le message d'être divisé en segments par le gestionnaire de files d'attente. Si elle est spécifiée pour un message qui est déjà un segment, cette option empêche le segment d'être divisé en segments plus petits.

La valeur de cet indicateur est zéro binaire. Il s'agit de l'option par défaut.

### MFSEGA

Segmentation autorisée.

Cette option permet au message d'être divisé en segments par le gestionnaire de files d'attente. Si elle est spécifiée pour un message qui est déjà un segment, cette option permet de le scinder en segments plus petits. MFSEGA peut être défini sans que MFSEG ou MFLSEG ne soit défini.

Lorsque le gestionnaire de files d'attente segmente un message, il active l'indicateur MFSEG dans la copie du MQMD envoyé avec chaque segment, mais ne modifie pas les paramètres de ces indicateurs dans le MQMD fourni par l'application sur l'appel MQPUT ou MQPUT1. Pour le dernier segment du message logique, le gestionnaire de files d'attente active également l'indicateur MFLSEG dans le MQMD envoyé avec le segment.

**Remarque :** Des précautions sont nécessaires lorsque des messages sont insérés avec MFSEGA mais sans PMLOGO. Si le message est:

- N'est pas un segment, et
- Pas dans un groupe, et
- Non réacheminé,

l'application doit se rappeler de réinitialiser la zone MDGID à GINONE avant chaque appel MQPUT ou MQPUT1, afin de générer un identificateur de groupe unique par le gestionnaire de files d'attente pour chaque message. Si tel n'est pas le cas, des messages non liés risquent de se retrouver par inadvertance avec le même identificateur de groupe, ce qui peut entraîner un traitement incorrect par la suite. Pour plus d'informations sur le moment où la zone MDGID doit être réinitialisée, voir les descriptions de la zone MDGID et de l'option PMLOGO.

Le gestionnaire de files d'attente divise les messages en segments selon les besoins afin de s'assurer que les segments (ainsi que les données d'en-tête éventuellement requises) correspondent à la file d'attente. Toutefois, il existe une limite inférieure pour la taille d'un segment généré par le gestionnaire de files d'attente, et seul le dernier segment créé à partir d'un message peut être inférieur à cette limite. (La limite inférieure pour la taille d'un segment généré par l'application est d'un octet.) Les segments générés par le gestionnaire de files d'attente peuvent être de longueur inégale. Le gestionnaire de files d'attente traite le message comme suit:

- Les formats définis par l'utilisateur sont divisés sur des limites multiples de 16 octets. Cela signifie que le gestionnaire de files d'attente ne génère pas de segments inférieurs à 16 octets (autres que le dernier segment).
- Les formats intégrés autres que le FMSTR sont divisés aux points appropriés à la nature des données présentes. Toutefois, le gestionnaire de files d'attente ne fractionne jamais un message au milieu d'une structure d'en-tête MQ. Cela signifie qu'un segment contenant une structure d'en-tête MQ unique ne peut pas être fractionné davantage par le gestionnaire de files d'attente et que, par conséquent, la taille de segment minimale possible pour ce message est supérieure à 16 octets.

Le deuxième segment ou un segment ultérieur généré par le gestionnaire de files d'attente commence par l'un des segments suivants:

- Une structure d'en-tête MQ
- Début des données de message d'application
- Une partie des données de message d'application
- FMSTR est divisé sans tenir compte de la nature des données présentes (SBCS, DBCS ou SBCS/DBCS mixte). Lorsque la chaîne est de type DBCS ou SBCS/DBCS mixte, il se peut que des segments ne puissent pas être convertis d'un jeu de caractères à un autre. Le gestionnaire de files d'attente ne fractionne jamais les messages FMSTR en segments de moins de 16 octets (autres que le dernier segment).
- Les zones MDFMT, MDCSI et MDENC du MQMD de chaque segment sont définies par le gestionnaire de files d'attente pour décrire correctement les données présentes au début du segment; le nom de format sera soit le nom d'un format intégré, soit le nom d'un format défini par l'utilisateur.
- La zone MDREP dans le MQMD des segments dont la valeur de MDOFF est supérieure à zéro est modifiée comme suit:
  - Pour chaque type de rapport, si l'option de rapport est RO\* D, mais que le segment ne peut pas contenir les 100 premiers octets de données utilisateur (c'est-à-dire les données suivant les structures d'en-tête MQ éventuellement présentes), l'option de rapport est remplacée par RO\*.

Le gestionnaire de files d'attente respecte les règles précédentes, mais fractionne les messages de manière imprévisible; ne faites pas d'hypothèses sur l'endroit où un message est fractionné

Pour les messages persistants, le gestionnaire de files d'attente peut effectuer une segmentation uniquement dans une unité de travail:

- Si l'appel MQPUT ou MQPUT1 s'exécute dans une unité de travail définie par l'utilisateur, cette unité de travail est utilisée. Si l'appel échoue à mi-chemin du processus de segmentation, le gestionnaire de files d'attente supprime tous les segments qui ont été placés dans la file d'attente suite à l'échec de l'appel. Toutefois, l'échec n'empêche pas la validation de l'unité de travail.

- Si l'appel est effectué en dehors d'une unité de travail définie par l'utilisateur et qu'il n'existe aucune unité de travail définie par l'utilisateur, le gestionnaire de files d'attente crée une unité de travail uniquement pour la durée de l'appel. Si l'appel aboutit, le gestionnaire de files d'attente valide automatiquement l'unité d'oeuvre (l'application n'a pas besoin de le faire). Si l'appel échoue, le gestionnaire de files d'attente annule l'unité d'oeuvre.
- Si l'appel est exécuté en dehors d'une unité de travail définie par l'utilisateur, mais qu'une unité de travail définie par l'utilisateur existe, le gestionnaire de files d'attente ne peut pas effectuer de segmentation. Si le message ne nécessite pas de segmentation, l'appel peut tout de même aboutir. Mais si le message requiert une segmentation, l'appel échoue avec le code anomalie RC2255.

Pour les messages non persistants, le gestionnaire de files d'attente ne nécessite pas qu'une unité d'oeuvre soit disponible pour effectuer la segmentation.

Une attention particulière doit être accordée à la conversion des données des messages qui peuvent être segmentés:

- Si la conversion de données est effectuée uniquement par l'application de réception sur l'appel MQGET et que l'application spécifie l'option GMCMPM, l'exit de conversion de données reçoit le message complet de l'exit à convertir et le fait que le message ait été segmenté n'apparaît pas à l'exit.
- Si l'application de réception extrait un segment à la fois, l'exit de conversion de données est appelé pour convertir un segment à la fois. La sortie doit donc être capable de convertir les données dans un segment indépendamment des données dans n'importe lequel des autres segments.

Si la nature des données du message est telle que la segmentation arbitraire des données sur des limites de 16 octets peut entraîner des segments qui ne peuvent pas être convertis par l'exit, ou si le format est FMSTR et que le jeu de caractères est DBCS ou SBCS/DBCS mixte, l'application émettrice doit elle-même créer et placer les segments, en spécifiant MFSEGI pour supprimer la segmentation. De cette manière, l'application émettrice peut s'assurer que chaque segment contient suffisamment d'informations pour permettre à l'exit de conversion de données de convertir le segment avec succès.

- Si la conversion de l'émetteur est indiquée pour un agent MCA, celui-ci convertit uniquement les messages qui ne sont pas des segments de messages logiques ; l'agent MCA ne tente jamais de convertir les messages qui sont des segments.

Cet indicateur est un indicateur d'entrée sur les appels MQPUT et MQPUT1 et un indicateur de sortie sur l'appel MQGET. Lors de ce dernier appel, le gestionnaire de files d'attente renvoie également la valeur de l'indicateur à la zone GMSEG dans MQGMO.

La valeur initiale de cet indicateur est MFSEGI.

**Indicateurs de statut:** indicateurs qui indiquent si le message physique appartient à un groupe de messages, s'il s'agit d'un segment d'un message logique, les deux ou aucun des deux. Un ou plusieurs des éléments suivants peuvent être spécifiés dans l'appel MQPUT ou MQPUT1 ou renvoyés par l'appel MQGET:

**MFMI**

Le message est membre d'un groupe.

**MFLMI**

Le message est le dernier message logique d'un groupe.

Si cet indicateur est défini, le gestionnaire de files d'attente active MFMI dans la copie de MQMD envoyée avec le message, mais ne modifie pas les paramètres de ces indicateurs dans le MQMD fourni par l'application sur l'appel MQPUT ou MQPUT1 .

Un groupe ne peut comporter qu'un seul message logique. Si tel est le cas, MFLMI est défini, mais la zone MDSEQ a la valeur un.

**MFSE**

Le message est un segment d'un message logique.

Lorsque MFSEG est spécifié sans MFLSEG, la longueur des données de message d'application dans le segment (à l'exclusion des longueurs des structures d'en-tête MQ qui peuvent être présentes) doit être au moins une. Si la longueur est égale à zéro, l'appel MQPUT ou MQPUT1 échoue avec le code anomalie RC2253.

### **MFLSEG**

Le message est le dernier segment d'un message logique.

Si cet indicateur est défini, le gestionnaire de files d'attente active MFSEG dans la copie de MQMD envoyée avec le message, mais ne modifie pas les paramètres de ces indicateurs dans le MQMD fourni par l'application sur l'appel MQPUT ou MQPUT1 .

Il est possible qu'un message logique ne comporte qu'un seul segment. Si tel est le cas, MFLSEG est défini, mais la zone MDOFF a la valeur zéro.

Lorsque MFLSEG est spécifié, la longueur des données de message d'application dans le segment (à l'exclusion des longueurs des structures d'en-tête éventuellement présentes) peut être égale à zéro.

L'application doit s'assurer que ces indicateurs sont correctement définis lors de l'insertion de messages. Si PMLOGO est spécifié ou a été spécifié dans l'appel MQPUT précédent pour l'identificateur de file d'attente, les paramètres des indicateurs doivent être cohérents avec les informations de groupe et de segment conservées par le gestionnaire de files d'attente pour l'identificateur de file d'attente. Les conditions suivantes s'appliquent aux appels MQPUT successifs pour le descripteur de file d'attente lorsque PMLOGO est spécifié:

- S'il n'y a pas de groupe ou de message logique en cours, tous ces indicateurs (et leurs combinaisons) sont valides.
- Une fois que MFMIG a été spécifié, il doit rester en fonction jusqu'à ce que MFLMIG soit spécifié. L'appel échoue avec le code anomalie RC2241 si cette condition n'est pas satisfaite.
- Une fois que MFSEG a été indiqué, il doit rester en fonction jusqu'à ce que MFLSEG soit indiqué. L'appel échoue avec le code anomalie RC2242 si cette condition n'est pas satisfaite.
- Une fois que MFSEG a été spécifié sans MFMIG, MFMIG doit rester hors fonction jusqu'à ce que MFLSEG ait été spécifié. L'appel échoue avec le code anomalie RC2242 si cette condition n'est pas satisfaite.

Le [tableau 1](#) présente les combinaisons valides des indicateurs et les valeurs utilisées pour les différents champs.

Ces indicateurs sont des indicateurs d'entrée sur les appels MQPUT et MQPUT1 et des indicateurs de sortie sur l'appel MQGET. Lors de ce dernier appel, le gestionnaire de files d'attente répercute également les valeurs des indicateurs dans les zones GMGST et GMSST de MQGMO.

**Indicateurs par défaut:** Vous pouvez indiquer ce qui suit pour indiquer que le message possède des attributs par défaut:

### **MFNONE**

Aucun indicateur de message (attributs de message par défaut).

Cela inhibe la segmentation et indique que le message ne fait pas partie d'un groupe et n'est pas un segment d'un message logique. MFNONE est défini pour faciliter la documentation du programme. Il n'est pas prévu que cet indicateur soit utilisé avec d'autres, mais comme sa valeur est nulle, une telle utilisation ne peut pas être détectée.

La zone MDMFL est partitionnée en sous-zones ; pour plus de détails, voir [«Options de rapport et indicateurs de message sous IBM i»](#), à la page 1505.

La valeur initiale de cette zone est MFNONE. Cette zone est ignorée si MDVER est inférieur à MDVER2.

### **MDMID (chaîne de 24 octets)**

Identificateur de message.

Il s'agit d'une chaîne d'octets utilisée pour distinguer un message d'un autre. En règle générale, deux messages ne doivent pas avoir le même identificateur de message, bien que cela ne soit pas interdit

par le gestionnaire de files d'attente. L'identificateur de message est une propriété permanente du message et est conservé lors des redémarrages du gestionnaire de files d'attente. Etant donné que l'identificateur de message est une chaîne d'octets et non une chaîne de caractères, l'identificateur de message n'est pas converti entre les jeux de caractères lorsque le message passe d'un gestionnaire de files d'attente à un autre.

Pour les appels MQPUT et MQPUT1, si MINONE ou PMNMID est spécifié par l'application, le gestionnaire de files d'attente génère un identificateur de message unique lors de l'insertion du message et le place dans le descripteur de message envoyé avec le message. Le gestionnaire de files d'attente renvoie également cet identificateur de message dans le descripteur de message appartenant à l'application émettrice. L'application peut utiliser cette valeur pour enregistrer des informations sur des messages particuliers et pour répondre à des requêtes provenant d'autres parties de l'application.

Un MDMID généré par le gestionnaire de files d'attente se compose d'un identificateur de produit de 4 octets (AMQ- ou CSQ- en ASCII ou EBCDIC, où - représente un caractère blanc unique), suivi d'une implémentation spécifique au produit d'une chaîne unique. Dans IBM MQ, il contient les 12 premiers caractères du nom du gestionnaire de files d'attente et une valeur dérivée de l'horloge système. Tous les gestionnaires de files d'attente qui peuvent intercommuniquer doivent donc avoir des noms qui diffèrent dans les 12 premiers caractères, afin de s'assurer que les identificateurs de message sont uniques. La possibilité de générer une chaîne unique dépend également du fait que l'horloge système n'est pas modifiée vers l'arrière. Pour éviter qu'un identificateur de message généré par le gestionnaire de files d'attente ne soit dupliqué par l'application, celle-ci doit éviter de générer des identificateurs dont les caractères initiaux sont compris entre A et I en ASCII ou EBCDIC (X'41'à X'49'et X'C1'à X'C9'). Toutefois, l'application n'est pas empêchée de générer des identificateurs avec des caractères initiaux dans ces plages.

Si le message est inséré dans une rubrique, le gestionnaire de files d'attente génère des identificateurs de message uniques pour chaque message publié. Si PMNMID est spécifié par l'application, le gestionnaire de files d'attente génère un identificateur de message unique à renvoyer en sortie. Si MINONE est spécifié par l'application, la valeur de la zone MDMID dans le MQMD n'est pas modifiée en cas de retour de l'appel.

Voir la description de PMRET dans [PMOPT](#) pour plus de détails sur les publications conservées.

Si le message est inséré dans une liste de distribution, le gestionnaire de files d'attente génère des identificateurs de message uniques si nécessaire, mais la valeur de la zone MDMID dans MQMD est inchangée lors du retour de l'appel, même si MINONE ou PMNMID a été spécifié. Si l'application a besoin de connaître les identificateurs de message générés par le gestionnaire de files d'attente, elle doit fournir des enregistrements MQPMR contenant la zone PRMID.

L'application émettrice peut également spécifier une valeur particulière pour l'identificateur de message, autre que MINONE ; cela empêche le gestionnaire de files d'attente de générer un identificateur de message unique. Une application qui transfère un message peut utiliser cette fonction pour propager l'identificateur du message d'origine.

Le gestionnaire de files d'attente n'utilise lui-même cette zone que pour :

- Générer une valeur unique si demandé, comme décrit précédemment
- Distribuer la valeur à l'application qui émet la demande d'obtention du message
- Copier la valeur dans la zone MDCID de tout message de rapport généré à propos de ce message (en fonction des options MDREP)

Lorsque le gestionnaire de files d'attente ou un agent MCA génère un message de rapport, il définit la zone MDMID de la manière indiquée par la zone MDREP du message d'origine, RONMI ou ROPMI. Les applications qui génèrent des messages de rapport doivent également effectuer cette opération.

Pour l'appel MQGET, MDMID est l'une des cinq zones qui peuvent être utilisées pour sélectionner un message particulier à extraire de la file d'attente. Normalement, l'appel MQGET renvoie le message suivant dans la file d'attente, mais si un message particulier est requis, il peut être obtenu en spécifiant un ou plusieurs des cinq critères de sélection, dans n'importe quelle combinaison ; ces zones sont les suivantes :

- MDMID
- MDCID
- MDGID
- MDSEQ
- MDOFF

L'application définit une ou plusieurs de ces zones sur les valeurs requises, puis définit les options de correspondance MO\* correspondantes dans la zone GMMO de MQGMO pour indiquer que ces zones doivent être utilisées comme critères de sélection. Seuls les messages dont les valeurs sont indiquées dans ces zones peuvent être extraits. La valeur par défaut de la zone GMMO (si elle n'est pas modifiée par l'application) correspond à la fois à l'identificateur de message et à l'identificateur de corrélation.

Normalement, le message renvoyé est le premier message de la file d'attente qui répond aux critères de sélection. Mais si GMBRWN est spécifié, le message renvoyé est le message suivant qui répond aux critères de sélection ; l'analyse de ce message commence par le message qui suit la position actuelle du curseur.

**Remarque :** La file d'attente est analysée séquentiellement à la recherche d'un message qui répond aux critères de sélection. Par conséquent, les temps d'extraction sont plus lents que si aucun critère de sélection n'est spécifié, en particulier si de nombreux messages doivent être analysés avant qu'un message approprié ne soit trouvé.

Voir [Tableau 1](#) pour plus d'informations sur la façon dont les critères de sélection sont utilisés dans diverses situations.

La spécification de MINONE comme identificateur de message a le même effet que la non-spécification de MOMSGI, c'est-à-dire que tout identificateur de message correspond.

Cette zone est ignorée si l'option GMMUC est spécifiée dans le paramètre **GMO** de l'appel MQGET.

En cas de retour d'un appel MQGET, la zone MDMID est définie sur l'identificateur du message renvoyé (le cas échéant).

La valeur spéciale suivante peut être utilisée:

**MINONE**

Aucun identificateur de message n'est indiqué.

La valeur est zéro binaire pour la longueur de la zone.

Il s'agit d'une zone d'entrée-sortie pour les appels MQGET, MQPUT et MQPUT1 . La longueur de cette zone est indiquée par LNMID. La valeur initiale de cette zone est MINONE.

**MDMT (entier signé à 10 chiffres)**

Type de message.

Indique le type du message. Les types de message sont regroupés comme suit:

**MTSFST**

Valeur la plus basse pour les types de message définis par le système.

**TMTSLST**

Valeur la plus élevée pour les types de message définis par le système.

Les valeurs suivantes sont actuellement définies dans la plage système:

**MTDGRM**

Message ne nécessitant pas de réponse.

Le message ne requiert pas de réponse.

**MTRQST**

Message nécessitant une réponse.

Le message requiert une réponse.



Le nom de la file d'attente à laquelle la réponse doit être envoyée doit être indiqué dans la zone MDRQ . La zone MDREP indique comment les valeurs MDMID et MDCID de la réponse doivent être définies.

#### **MTRPLY**

Répondre à un message de demande antérieur.

Le message correspond à la réponse à un message de demande antérieur (MTRQST). Le message doit être envoyé à la file d'attente indiquée par la zone MDRQ du message de demande. La zone MDREP de la demande doit être utilisée pour contrôler la façon dont les valeurs MDMID et MDCID de la réponse sont définies.

**Remarque :** Le gestionnaire de files d'attente n'applique pas la relation de demande / réponse ; il s'agit d'une responsabilité d'application.

#### **MTRPRT**

Message de rapport.

Le message signale une occurrence attendue ou inattendue, généralement liée à un autre message (par exemple, un message de demande a été reçu qui contenait des données non valides). Le message doit être envoyé à la file d'attente indiquée par la zone MDRQ du descripteur de message du message d'origine. La zone MDFB doit être définie pour indiquer la nature du rapport. La zone MDREP du message d'origine peut être utilisée pour contrôler la manière dont les valeurs MDMID et MDCID du message de rapport doivent être définies.

Les messages de rapport générés par le gestionnaire de files d'attente ou l'agent de canal de transmission de messages sont toujours envoyés à la file d'attente MDRQ , avec les zones MDFB et MDCID définies comme décrit précédemment.

D'autres valeurs de la plage système peuvent être définies dans les versions futures de l'interface MQI et sont acceptées par les appels MQPUT et MQPUT1 sans erreur.

Les valeurs définies par l'application peuvent également être utilisées. Ils doivent être compris dans la plage suivante:

#### **MTAFST**

Valeur la plus faible pour les types de message définis par l'application.

#### **MTALST**

Valeur la plus élevée pour les types de message définis par l'application.

Pour les appels MQPUT et MQPUT1 , la valeur MDMT doit être comprise dans la plage définie par le système ou dans la plage définie par l'application. Si ce n'est pas le cas, l'appel échoue avec le code anomalie RC2029.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La valeur initiale de cette zone est MTDGRM.

#### **MDOFF (entier signé à 10 chiffres)**

Décalage des données dans le message physique à partir du début du message logique.

Il s'agit du décalage en octets des données du message physique à partir du début du message logique dont font partie les données. Ces données sont appelées un *segment*. Le décalage est compris entre 0 et 999 999 999. Un message physique qui n'est pas un segment d'un message logique a un décalage de zéro.

Cette zone n'a pas besoin d'être définie par l'application sur l'appel MQPUT ou MQGET si:

- Dans l'appel MQPUT, PMLOGO est spécifié.
- Dans l'appel MQGET, MOOFFS n'est pas spécifié.

Il s'agit des méthodes recommandées pour utiliser ces appels pour les messages qui ne sont pas des messages de rapport. Toutefois, si l'application ne respecte pas ces conditions ou si l'appel est MQPUT1, l'application doit s'assurer que MDOFF est défini sur une valeur appropriée.

Lors de l'entrée dans les appels MQPUT et MQPUT1, le gestionnaire de files d'attente utilise la valeur détaillée dans le [Tableau 1](#). Dans la sortie des appels MQPUT et MQPUT1, le gestionnaire de files d'attente définit cette zone sur la valeur qui a été envoyée avec le message.

Pour un rapport de message de rapport sur un segment d'un message logique, la zone MDOLN (à condition qu'elle ne soit pas OLUNDF) est utilisée pour mettre à jour le décalage dans les informations de segment conservées par le gestionnaire de files d'attente.

En entrée de l'appel MQGET, le gestionnaire de files d'attente utilise la valeur détaillée dans le [Tableau 1](#). Dans la sortie de l'appel MQGET, le gestionnaire de files d'attente définit cette zone sur la valeur du message extrait.

La valeur initiale de cette zone est zéro. Cette zone est ignorée si MDVER est inférieur à MDVER2.

### **MDOLN (entier signé à 10 chiffres)**

Longueur du message d'origine.

Cette zone est pertinente uniquement pour les messages de rapport qui sont des segments. Elle indique la longueur du segment de message auquel se rapporte le message de rapport ; elle n'indique pas la longueur du message logique dont fait partie le segment, ni la longueur des données du message de rapport.

**Remarque :** Lors de la génération d'un message de rapport pour un message qui est un segment, le gestionnaire de files d'attente et l'agent de canal de message sont copiés dans le MQMD pour le message de rapport dans les zones MDGID, MDSEQ, MDOFFet *MDMFL* du message d'origine. Par conséquent, le message de rapport est également un segment. Il est recommandé aux applications qui génèrent des messages de rapport de faire de même et de s'assurer que la zone MDOLN est définie correctement.

La valeur spéciale suivante est définie:

#### **ANNULER**

Longueur d'origine du message non définie.

MDOLN est une zone d'entrée dans les appels MQPUT et MQPUT1, mais la valeur fournie par l'application n'est acceptée que dans des circonstances particulières:

- Si le message en cours d'insertion est un segment et est également un message de rapport, le gestionnaire de files d'attente accepte la valeur spécifiée. La valeur doit être:
  - Supérieur à zéro si le segment n'est pas le dernier segment
  - Pas moins de zéro si le segment est le dernier segment
  - Pas moins que la longueur des données présentes dans le message

Si ces conditions ne sont pas satisfaites, l'appel échoue avec le code anomalie RC2252.

- Si le message en cours d'insertion est un segment mais pas un message de rapport, le gestionnaire de files d'attente ignore la zone et utilise à la place la longueur des données de message d'application.
- Dans tous les autres cas, le gestionnaire de files d'attente ignore la zone et utilise la valeur OLUNDF à la place.

Il s'agit d'une zone de sortie de l'appel MQGET.

La valeur initiale de cette zone est OLUNDF. Cette zone est ignorée si MDVER est inférieur à MDVER2.

### **MDPAN (chaîne de caractères de 28 octets)**

Nom de l'application qui a inséré le message.

Il fait partie du *contexte d'origine* du message. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

Le format de MDPAN dépend de la valeur de MDPAT.

Lorsque cette zone est définie par le gestionnaire de files d'attente (c'est-à-dire pour toutes les options à l'exception de PMSETA), elle est définie sur une valeur qui est déterminée par l'environnement:

- **z/OS** Sous z/OS, le gestionnaire de files d'attente utilise:
  - Pour le lot z/OS , nom de travail à 8 caractères provenant de la carte de travail JES
  - Pour TSO, l'ID utilisateur TSO à 7 caractères
  - Pour CICS, l'ID d'application à 8 caractères, suivi de l'ID de transaction à 4 caractères
  - Pour IMS, l'identificateur système IMS à 8 caractères, suivi du nom PSB à 8 caractères
  - Pour XCF, nom de groupe XCF à 8 caractères, suivi du nom de membre XCF à 16 caractères
  - Pour un message généré par un gestionnaire de files d'attente, les 28 premiers caractères du nom du gestionnaire de files d'attente
  - Pour la mise en file d'attente répartie sans CICS, nom de travail à 8 caractères de l'initiateur de canal, suivi du nom à 8 caractères du module placé dans la file d'attente de rebut, suivi d'un identificateur de tâche à 8 caractères.
  - Pour le traitement des liaisons de langage MQSeries Java avec IBM MQ for z/OS le nom de travail à 8 caractères de l'espace adresse créé pour l'environnement z/OS UNIX System Services . En règle générale, il s'agit d'un ID utilisateur TSO avec un seul caractère numérique ajouté.

Le ou les noms sont chacun complétés à droite par des blancs, comme n'importe quel espace dans le reste de la zone. Lorsqu'il y a plusieurs noms, il n'y a pas de séparateur entre eux.

- **Windows** Sur les systèmes PC DOS et Windows , le gestionnaire de files d'attente utilise:
  - Pour une application CICS , le nom de transaction CICS
  - Pour une application nonCICS , les 28 caractères les plus à droite du nom qualifié complet de l'exécutable
- **IBM i** Sous IBM i, le gestionnaire de files d'attente utilise le nom de travail qualifié complet.
- **Linux** **AIX** Sous AIX and Linux, le gestionnaire de files d'attente utilise:
  - Pour une application CICS , le nom de transaction CICS
  - Pour une application nonCICS , les 14 caractères les plus à droite du nom qualifié complet de l'exécutable si celui-ci est disponible pour le gestionnaire de files d'attente, et les blancs dans le cas contraire (par exemple, sous AIX)
- Sous VSE/ESA, le gestionnaire de files d'attente utilise l'ID application à 8 caractères, suivi de l'ID transaction à 4 caractères.

Pour les appels MQPUT et MQPUT1 , il s'agit d'une zone d'entrée-sortie si PMSETA est spécifié dans le paramètre **PMO** . Toutes les informations qui suivent un caractère null dans la zone sont supprimées. Le caractère null et tous les caractères suivants sont convertis en blancs par le gestionnaire de files d'attente. Si PMSETA n'est pas indiqué, cette zone est ignorée en entrée et est une zone de sortie uniquement.

Il s'agit d'une zone de sortie pour l'appel MQGET. La longueur de cette zone est indiquée par LNPAN. La valeur initiale de cette zone est de 28 caractères blancs.

### **MDPAT (entier signé à 10 chiffres)**

Type d'application qui a inséré le message.

Il fait partie du **contexte d'origine** du message. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

*MDPAT* peut avoir l'un des types standard suivants. Les types définis par l'utilisateur peuvent également être utilisés, mais ils doivent être limités aux valeurs comprises entre ATUFST et ATULST.

#### **ATAIX**

Application AIX (même valeur que ATUNIX).

**ATBRKR**

Courtier.

**ATCICS**

Transaction CICS .

**ATCICB**

CICS bridge.

**ATVSE**

Transaction CICS/VSE .

**ATDOS**

Application IBM MQ MQI client sur PC DOS.

**ATDQM**

Agent de gestionnaire de files d'attente réparties.

**ATGUAR**

Application Tandem Guardian (même valeur que ATNSK).

**ATIMS**

Application IMS .

**ATIMSB**

Passerelle IMS .

**ATJAVA**

Java.

**ATMVS**

Application MVS ou TSO (même valeur que ATZOS).

**REMARQUE ATNOTE**

Application de l'agent Lotus Notes .

**ATNSK**

Application Noyau NonStop en tandem.

**AT390**

Application OS/390 (même valeur que ATZOS).

**AT400**

Application IBM i .

**ATQM**

Gestionnaire de files d'attente.

**ATUNIX**

Application UNIX .

**ATVOS**

Application Stratus VOS.

**ATWIN**

Application Windows 16 bits.

**ATWINT**

Application Windows 32 bits.

**ATXCF**

XCF.

**ATZOS**

Application z/OS .

**ATDEF**

Type d'application par défaut.

Il s'agit du type d'application par défaut pour la plateforme sur laquelle l'application s'exécute.

**Remarque :** La valeur de cette constante est spécifique à l'environnement.

**ATUNK**

Type d'application inconnu.

Cette valeur peut être utilisée pour indiquer que le type d'application est inconnu, même si d'autres informations de contexte sont présentes.

**TATUFST**

Valeur la plus faible pour le type d'application défini par l'utilisateur.

**ATULST**

Valeur la plus élevée pour le type d'application défini par l'utilisateur.

La valeur spéciale suivante peut également se produire:

**ATNCON**

Aucune information contextuelle n'est présente dans le message.


Cette valeur est définie par le gestionnaire de files d'attente lorsqu'un message est inséré sans contexte (c'est-à-dire que l'option de contexte PMNOC est spécifiée).

Lorsqu'un message est extrait, MDPAT peut être testé pour cette valeur afin de déterminer si le message a un contexte (il est recommandé que MDPAT ne soit jamais défini sur ATNCON, par une application utilisant PMSETA, si l'une des autres zones de contexte n'est pas vide).

**ATSIB**

Indique un message provenant d'un autre produit de messagerie IBM MQ et arrivé via le pont SIB (Service Integration Bus).

Lorsque le gestionnaire de files d'attente génère ces informations à la suite d'une insertion d'application, la zone est définie sur une valeur déterminée par l'environnement.

 Notez que sous IBM i, la zone est définie sur AT400; le gestionnaire de files d'attente n'utilise jamais ATCICS sous IBM i.

Pour les appels MQPUT et MQPUT1, il s'agit d'une zone d'entrée-sortie si PMSETA est spécifié dans le paramètre **PMO**. Si PMSETA n'est pas indiqué, cette zone est ignorée en entrée et est une zone de sortie uniquement.

Une fois qu'un appel MQPUT ou MQPUT1 a abouti, cette zone contient le MDPAT qui a été transmis avec le message s'il a été inséré dans une file d'attente. Il s'agit de la valeur de MDPAT qui est conservée avec le message s'il est conservé (voir la description de PMRET pour plus de détails sur les publications conservées) mais qui n'est pas utilisée en tant que MDPAT lorsque le message est envoyé en tant que publication aux abonnés car ils fournissent une valeur pour remplacer MDPAT dans toutes les publications qui leur sont envoyées. Si le message n'a pas de contexte, la zone est définie sur ATNCON.

Il s'agit d'une zone de sortie pour l'appel MQGET. La valeur initiale de cette zone est ATNCON.

**MDPD (chaîne de caractères de 8 octets)**

Date à laquelle le message a été inséré.

Il fait partie du *contexte d'origine* du message. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

Le format utilisé pour la date à laquelle cette zone est générée par le gestionnaire de files d'attente est le suivant:

• AAAAMMJJ

où les caractères représentent:

**AAAA**

année (quatre chiffres)

**MM**

mois de l'année (01 à 12)

## **JJ**

jour du mois (01 à 31)

Le temps moyen de Greenwich (GMT) est utilisé pour les zones MDPD et MDPT , à condition que l'horloge système soit définie avec précision sur le temps moyen de Greenwich.

Si le message a été inséré dans une unité de travail, la date est celle à laquelle le message a été inséré et non celle à laquelle l'unité de travail a été validée.

Pour les appels MQPUT et MQPUT1 , il s'agit d'une zone d'entrée-sortie si PMSETA est spécifié dans le paramètre **PMO** . Le contenu de la zone n'est pas vérifié par le gestionnaire de files d'attente, sauf que les informations qui suivent un caractère null dans la zone sont supprimées. Le caractère null et tous les caractères suivants sont convertis en blancs par le gestionnaire de files d'attente. Si PMSETA n'est pas indiqué, cette zone est ignorée en entrée et est une zone de sortie uniquement.

Une fois qu'un appel MQPUT ou MQPUT1 a abouti, cette zone contient le MDPD qui a été transmis avec le message s'il a été inséré dans une file d'attente. Il s'agit de la valeur de MDPD qui est conservée avec le message s'il est conservé (voir la description de PMRET pour plus de détails sur les publications conservées) mais qui n'est pas utilisée en tant que MDPD lorsque le message est envoyé en tant que publication aux abonnés car ils fournissent une valeur pour remplacer MDPD dans toutes les publications qui leur sont envoyées. Si le message n'a pas de contexte, la zone est entièrement vide.

Il s'agit d'une zone de sortie pour l'appel MQGET. La longueur de cette zone est indiquée par LNPDAT. La valeur initiale de cette zone est de 8 caractères blancs.

### **MDPER (entier signé à 10 chiffres)**

Persistance des messages.

Indique si le message survit aux échecs du système et aux redémarrages du gestionnaire de files d'attente. Pour les appels MQPUT et MQPUT1 , la valeur doit être l'une des suivantes:

#### **PEPER**

Le message est persistant.

Cela signifie que le message survit aux échecs du système et aux redémarrages du gestionnaire de files d'attente. Une fois que le message a été inséré et que l'unité de travail du putter a été validée (si le message est inséré dans une unité de travail), le message est conservé dans la mémoire secondaire. Il y reste jusqu'à ce que le message soit supprimé de la file d'attente et que l'unité de travail de la méthode d'accès get soit validée (si le message est extrait dans le cadre d'une unité de travail).

Lorsqu'un message persistant est envoyé à une file d'attente éloignée, un mécanisme de stockage et de transfert est utilisé pour stocker le message sur chaque gestionnaire de files d'attente le long de la route menant à la destination, jusqu'à ce que le message soit connu comme étant arrivé sur le gestionnaire de files d'attente suivant.

Les messages persistants ne peuvent pas être placés sur:

- Files d'attente dynamiques temporaires
- Files d'attente partagées dans lesquelles le niveau de la structure d'unité de couplage est inférieur à trois ou dans lesquelles la structure d'unité de couplage n'est pas récupérable.

Les messages persistants peuvent être placés dans des files d'attente dynamiques permanentes, des files d'attente prédéfinies et des files d'attente partagées où le niveau de structure de l'unité de couplage est 3 et où l'unité de couplage est récupérable.

#### **PENPER**

Le message n'est pas persistant.

Cela signifie que le message ne survit normalement pas aux défaillances du système ou aux redémarrages du gestionnaire de files d'attente. Cela s'applique même si une copie intacte du message est trouvée sur la mémoire secondaire lors du redémarrage du gestionnaire de files d'attente.

Dans le cas spécial des files d'attente partagées, les messages non persistants *survivent* aux redémarrages des gestionnaires de files d'attente dans le groupe de partage de files d'attente, mais ne survivent pas aux échecs de l'unité de couplage utilisée pour stocker les messages dans les files d'attente partagées.

### PEQDEF

Le message a une persistance par défaut.

- Si la file d'attente est une file d'attente de cluster, la persistance du message est extraite de l'attribut **DefPersistence** défini sur le gestionnaire de files d'attente de destination qui possède l'instance particulière de la file d'attente dans laquelle le message est placé. Généralement, toutes les instances d'une file d'attente de cluster ont la même valeur pour l'attribut **DefPersistence**, bien que cela ne soit pas obligatoire.

La valeur de **DefPersistence** est copiée dans la zone *MDPER* lorsque le message est placé dans la file d'attente de destination. Si **DefPersistence** est modifié ultérieurement, les messages qui ont déjà été placés dans la file d'attente ne sont pas affectés.

- Si la file d'attente n'est pas une file d'attente de cluster, la persistance du message est extraite de l'attribut **DefPersistence** défini sur le gestionnaire de files d'attente local, même si le gestionnaire de files d'attente de destination est distant.

S'il existe plusieurs définitions dans le chemin de résolution de nom de file d'attente, la persistance par défaut est extraite de la valeur de cet attribut dans la première définition du chemin. Possibilités :

- Une file d'attente alias
- Une file d'attente locale
- Définition locale d'une file d'attente éloignée
- Un alias de gestionnaire de files d'attente
- Une file d'attente de transmission (par exemple, la file d'attente *DefXmitQName*)

La valeur de **DefPersistence** est copiée dans la zone *MDPER* lorsque le message est inséré. Si **DefPersistence** est modifié par la suite, les messages qui ont déjà été insérés ne sont pas affectés.

Les messages persistants et non persistants peuvent exister dans la même file d'attente.

Lors de la réponse à un message, les applications doivent normalement utiliser pour le message de réponse la persistance du message de demande.

Pour un appel *MQGET*, la valeur renvoyée est *PEPER* ou *PENPER*.

Il s'agit d'une zone de sortie pour l'appel *MQGET* et d'une zone d'entrée pour les appels *MQPUT* et *MQPUT1*. La valeur initiale de cette zone est *PEQDEF*.

### MDPRI (entier signé à 10 chiffres)

Priorité du message.

Pour les appels *MQPUT* et *MQPUT1*, la valeur doit être supérieure ou égale à zéro ; zéro est la priorité la plus basse. La valeur spéciale suivante peut également être utilisée:

### PRQDEF

Priorité par défaut pour la file d'attente.

- Si la file d'attente est une file d'attente de cluster, la priorité du message est extraite de l'attribut **DefPriority** défini au niveau du gestionnaire de files d'attente de destination qui possède l'instance particulière de la file d'attente dans laquelle le message est placé. Généralement, toutes les instances d'une file d'attente de cluster ont la même valeur pour l'attribut **DefPriority**, bien que cela ne soit pas obligatoire.

La valeur de **DefPriority** est copiée dans la zone MDPRI lorsque le message est placé dans la file d'attente de destination. Si **DefPriority** est modifié ultérieurement, les messages qui ont déjà été placés dans la file d'attente ne sont pas affectés.

- Si la file d'attente n'est pas une file d'attente de cluster, la priorité du message est extraite de l'attribut **DefPriority** défini sur le gestionnaire de files d'attente local, même si le gestionnaire de files d'attente de destination est distant.

S'il existe plusieurs définitions dans le chemin de résolution de nom de file d'attente, la priorité par défaut est extraite de la valeur de cet attribut dans la première définition du chemin.

Possibilités :

- Une file d'attente alias
- Une file d'attente locale
- Définition locale d'une file d'attente éloignée
- Un alias de gestionnaire de files d'attente
- Une file d'attente de transmission (par exemple, la file d'attente DefXmitQName)

La valeur de **DefPriority** est copiée dans la zone MDPRI lorsque le message est inséré. Si **DefPriority** est modifié par la suite, les messages qui ont déjà été insérés ne sont pas affectés.

La valeur renvoyée par l'appel MQGET est toujours supérieure ou égale à zéro ; la valeur PRQDEF n'est jamais renvoyée.

Si un message est inséré avec une priorité supérieure au maximum pris en charge par le gestionnaire de files d'attente local (ce maximum est donné par l'attribut de gestionnaire de files d'attente **MaxPriority**), le message est accepté par le gestionnaire de files d'attente, mais placé dans la file d'attente avec la priorité maximale du gestionnaire de files d'attente ; l'appel MQPUT ou MQPUT1 se termine avec CCWARN et le code anomalie RC2049. Toutefois, la zone MDPRI conserve la valeur spécifiée par l'application qui a inséré le message.

Lors de la réponse à un message, les applications doivent normalement utiliser pour le message de réponse la priorité du message de demande. Dans d'autres cas, la spécification de PRQDEF permet d'optimiser les priorités sans modifier l'application.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1. La valeur initiale de cette zone est PRQDEF.

### MDPT (chaîne de caractères de 8 octets)

Heure à laquelle le message a été inséré.

Il fait partie du **contexte d'origine** du message. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

Le format utilisé pour l'heure à laquelle cette zone est générée par le gestionnaire de files d'attente est le suivant:

- HHMMSSSTH

où les caractères représentent (dans l'ordre):

**HH**

heures (00 à 23)

**MM**

minutes (00 à 59)

**SS**

secondes (00 à 59 ; voir la [remarque](#))

**T**

dixièmes de seconde (0 à 9)

**H**

centièmes de seconde (0 à 9)



**Remarque :** Si l'horloge système est synchronisée selon une norme de temps très précise, il est possible en de rares occasions que 60 ou 61 soient renvoyés pour les secondes dans MDPT. Cela se produit lorsque des secondes bissextiles sont insérées dans la norme de temps globale.

Le temps moyen de Greenwich (GMT) est utilisé pour les zones MDPD et MDPT, à condition que l'horloge système soit définie avec précision sur le temps moyen de Greenwich.

Si le message a été inséré dans une unité de travail, l'heure est celle à laquelle le message a été inséré et non celle à laquelle l'unité de travail a été validée.

Pour les appels MQPUT et MQPUT1, il s'agit d'une zone d'entrée-sortie si PMSETA est spécifié dans le paramètre **PMO**. Le contenu de la zone n'est pas vérifié par le gestionnaire de files d'attente, sauf que les informations qui suivent un caractère null dans la zone sont supprimées. Le caractère null et tous les caractères suivants sont convertis en blancs par le gestionnaire de files d'attente. Si PMSETA n'est pas indiqué, cette zone est ignorée en entrée et est une zone de sortie uniquement.

Une fois qu'un appel MQPUT ou MQPUT1 a abouti, cette zone contient la valeur MDPT qui a été transmise avec le message si elle a été placée dans une file d'attente. Il s'agit de la valeur de MDPT qui est conservée avec le message s'il est conservé (voir la description de PMRET pour plus de détails sur les publications conservées) mais qui n'est pas utilisée en tant que MDPT lorsque le message est envoyé en tant que publication aux abonnés car ils fournissent une valeur pour remplacer MDPT dans toutes les publications qui leur sont envoyées. Si le message n'a pas de contexte, la zone est entièrement vide.

Il s'agit d'une zone de sortie pour l'appel MQGET. La longueur de cette zone est indiquée par LNPTIM. La valeur initiale de cette zone est de 8 caractères blancs.

### **MDREP (entier signé à 10 chiffres)**

Options des messages de rapport.

Un message de rapport est un message relatif à un autre message, utilisé pour informer une application des événements attendus ou inattendus liés au message d'origine. La zone MDREP permet à l'application qui envoie le message d'origine de spécifier quels sont les messages de rapport requis, si les données de message d'application doivent y être incluses, ainsi que (pour les rapports et les réponses) la manière dont les identificateurs de message et de corrélation du message de rapport ou de réponse doivent être définis. Tout ou partie (ou aucun) des types de message de rapport suivants peut être demandé:

- Exception
- Expiration
- Confirmer à l'arrivée (COA)
- Confirmer à la livraison (COD)
- Notification d'action positive (PAN)
- Notification d'action négative (NAN)

Si plusieurs types de message de rapport sont requis, ou si d'autres options de rapport sont nécessaires, les valeurs peuvent être ajoutées ensemble (n'ajoutez pas la même constante plusieurs fois).

L'application qui reçoit le message de rapport peut déterminer la raison pour laquelle le rapport a été généré en examinant la zone MDFB dans le MQMD; voir la zone MDFB pour plus de détails.

L'utilisation des options de rapport lors de l'insertion d'un message dans une rubrique peut entraîner la génération de zéro, un ou plusieurs messages de rapport et leur envoi à l'application. En effet, le message de publication peut être envoyé à zéro, une ou plusieurs applications d'abonnement.

**Options d'exception:** Vous pouvez spécifier l'une des options suivantes pour demander un message de rapport d'exception.

### **ROACTIVITE**

Rapports d'activité requis

Cette option de rapport permet de générer un rapport d'activité chaque fois qu'un message avec ce jeu d'options de rapport est traité par des applications de support.

Les messages avec ce jeu d'options de rapport doivent être acceptés par n'importe quel gestionnaire de files d'attente, même s'ils ne comprennent pas l'option. Cela permet de définir l'option de rapport sur n'importe quel message utilisateur, même s'ils sont traités par les gestionnaires de files d'attente précédents. Pour ce faire, l'option de rapport est placée dans la sous-zone ROAUM.

Si un processus (un gestionnaire de files d'attente ou un processus utilisateur) effectue une activité sur un message avec ROACT défini, il peut choisir de générer et d'insérer un rapport d'activité.

L'option de rapport d'activités permet de tracer la route de tout message dans un réseau de gestionnaires de files d'attente. L'option de rapport peut être spécifiée sur n'importe quel message utilisateur en cours et elle peut commencer instantanément à calculer la route du message via le réseau. Si l'application générant le message ne peut pas activer la génération de rapport d'activité, elle peut être activée à l'aide d'un exit de croisement d'API fourni par les administrateurs du gestionnaire de files d'attente.

Plusieurs conditions sont applicables aux rapports d'activité:

1. La route sera moins détaillée s'il y a moins de gestionnaires de files d'attente dans le réseau qui sont en mesure de générer des rapports d'activité.
2. Les rapports d'activité peuvent ne pas être facilement "commandables" pour déterminer la route empruntée.
3. Il se peut que les rapports d'activité ne puissent pas trouver de route vers leur destination demandée.

## **ROEXC**

Rapports d'exception requis.

Ce type de rapport peut être généré par un agent MCA lorsqu'un message est envoyé à un autre gestionnaire de files d'attente et que le message ne peut pas être distribué à la file d'attente de destination spécifiée. Par exemple, la file d'attente de destination ou une file d'attente de transmission intermédiaire peut être saturée ou le message peut être trop volumineux pour la file d'attente.

La génération du message de rapport d'exception dépend de la persistance du message d'origine et de la vitesse du canal de transmission des messages (normal ou rapide) par lequel le message d'origine est transmis:

- Pour tous les messages persistants et pour les messages non persistants qui transitent par des canaux de messages normaux, le rapport d'exception est généré uniquement si l'action spécifiée par l'application émettrice pour la condition d'erreur peut aboutir. L'application émettrice peut spécifier l'une des actions suivantes pour contrôler la disposition du message d'origine lorsque la condition d'erreur se produit:
  - RODLQ (le message d'origine est placé dans la file d'attente des messages non livrés).
  - RODISC (ce qui entraîne la suppression du message d'origine).

Si l'action spécifiée par l'application émettrice ne peut pas aboutir, le message d'origine est laissé dans la file d'attente de transmission et aucun message de rapport d'exception n'est généré.

- Pour les messages non persistants qui transitent par des canaux de messages rapides, le message d'origine est supprimé de la file d'attente de transmission et le rapport d'exception est généré même si l'action spécifiée pour la condition d'erreur ne peut pas aboutir. Par exemple, si RODLQ est indiqué, mais que le message d'origine ne peut pas être placé dans la file d'attente de rebut car (par exemple) cette file d'attente est saturée, le message de rapport d'exception est généré et le message d'origine est supprimé.

Pour plus d'informations sur les canaux de message normaux et rapides, voir [Persistance des messages](#) .

Aucun rapport d'exception n'est généré si l'application qui a inséré le message d'origine peut être avertie de manière synchrone de l'incident au moyen du code anomalie renvoyé par l'appel MQPUT ou MQPUT1 .

Les applications peuvent également envoyer des rapports d'exception pour indiquer qu'un message qu'elles ont reçu ne peut pas être traité (par exemple, parce qu'il s'agit d'une transaction de débit qui entraînerait le dépassement de la limite de crédit du compte).

Les données du message d'origine ne sont pas incluses dans le message de rapport.

N'indiquez pas plus d'un ROEXC, ROEXCD ou ROEXCF.

#### **ROEXCD**

Des rapports d'exception contenant des données sont requis.

Cette valeur est identique à ROEXC, sauf que les 100 premiers octets des données de message d'application du message d'origine sont inclus dans le message de rapport. Si le message d'origine contient une ou plusieurs structures d'en-tête MQ , elles sont incluses dans le message de rapport, en plus des 100 octets de données d'application.

N'indiquez pas plus d'un ROEXC, ROEXCD ou ROEXCF.

#### **ROEXCF**

Des rapports d'exception avec des données complètes sont requis.

Cette valeur est identique à ROEXC, sauf que toutes les données de message d'application du message d'origine sont incluses dans le message de rapport.

N'indiquez pas plus d'un ROEXC, ROEXCD ou ROEXCF.

**Options d'expiration:** Vous pouvez spécifier l'une des options suivantes pour demander un message de rapport d'expiration.

#### **ROEXP**

Rapports d'expiration requis.

Ce type de rapport est généré par le gestionnaire de files d'attente si le message est supprimé avant la livraison à une application en raison du délai d'expiration (voir la zone MDEXP ). Si cette option n'est pas définie, aucun message de rapport n'est généré si un message est supprimé pour cette raison (même si l'une des options ROEXC\* est spécifiée).

Les données du message d'origine ne sont pas incluses dans le message de rapport.

N'indiquez pas plus d'un ROEXP, ROEXPD ou ROEXPF.

#### **ROEXPD**

Rapports d'expiration avec données requises.

Cette valeur est identique à ROEXP, sauf que les 100 premiers octets des données de message d'application du message d'origine sont inclus dans le message de rapport. Si le message d'origine contient une ou plusieurs structures d'en-tête MQ , elles sont incluses dans le message de rapport, en plus des 100 octets de données d'application.

N'indiquez pas plus d'un ROEXP, ROEXPD ou ROEXPF.

#### **ROEXPF**

Des rapports d'expiration avec des données complètes sont requis.

Cette valeur est identique à ROEXP, sauf que toutes les données de message d'application du message d'origine sont incluses dans le message de rapport.

N'indiquez pas plus d'un ROEXP, ROEXPD ou ROEXPF.

**Options de confirmation à l'arrivée:** Vous pouvez spécifier l'une des options suivantes pour demander un message de rapport de confirmation à l'arrivée.

## **ROCOA**

Rapports de confirmation à l'arrivée requis.

Ce type de rapport est généré par le gestionnaire de files d'attente propriétaire de la file d'attente de destination, lorsque le message est placé dans la file d'attente de destination. Les données du message d'origine ne sont pas incluses dans le message de rapport.

Si le message est inséré dans une unité de travail et que la file d'attente de destination est une file d'attente locale, le message de rapport COA généré par le gestionnaire de files d'attente ne peut être extrait que si et lorsque l'unité de travail est validée.

Aucun rapport COA n'est généré si la zone MDFMT du descripteur de message est FMXQH ou FMDLH. Cela empêche la génération d'un rapport COA si le message est placé dans une file d'attente de transmission ou s'il n'est pas livrable et placé dans une file d'attente de rebut.

Ne spécifiez pas plus d'un ROCOA, ROCOAD ou ROCOAF.

## **ROCOAD**

Rapports de confirmation à l'arrivée avec les données requises.

Cette valeur est identique à ROCOA, sauf que les 100 premiers octets des données de message d'application du message d'origine sont inclus dans le message de rapport. Si le message d'origine contient une ou plusieurs structures d'en-tête MQ, elles sont incluses dans le message de rapport, en plus des 100 octets de données d'application.

Ne spécifiez pas plus d'un ROCOA, ROCOAD ou ROCOAF.

## **ROCOAF**

Rapports de confirmation à l'arrivée avec données complètes requises.

Cette valeur est identique à ROCOA, sauf que toutes les données de message d'application du message d'origine sont incluses dans le message de rapport.

Ne spécifiez pas plus d'un ROCOA, ROCOAD ou ROCOAF.

**Options de suppression et d'expiration:** Vous pouvez spécifier l'option suivante pour définir l'heure d'expiration et l'indicateur de suppression des messages de rapport.

## **ROPDAE**

Définissez l'heure d'expiration du message de rapport et l'indicateur de suppression.

Cette option garantit que les messages de rapport et les messages de réponse héritent de l'heure d'expiration et de l'indicateur de suppression (qu'ils soient ou non supprimés) de leurs messages d'origine. Lorsque cette option est définie, les messages de rapport et de réponse sont les suivants:

1. Hérite de l'indicateur RODISC (s'il a été défini).
2. Hérite de l'heure d'expiration restante du message, si le message n'est pas un rapport d'expiration. Si le message est un rapport d'expiration, le délai d'expiration est de 60 secondes.

Si cette option est définie, les conditions suivantes s'appliquent:

### **Remarque :**

1. Les messages de rapport et de réponse sont générés avec un indicateur de suppression et une valeur d'expiration et ne peuvent pas rester dans le système.
2. Les messages de route de trace ne peuvent pas atteindre les files d'attente de destination sur les gestionnaires de files d'attente activés pour les routes de trace.
3. Les files d'attente ne peuvent pas être remplies avec des rapports qui ne peuvent pas être distribués si les liaisons de communication sont rompues.
4. Les réponses du serveur de commandes héritent de l'expiration restante de la demande.

**Options de confirmation de distribution:** Vous pouvez spécifier l'une des options suivantes pour demander un message de rapport de confirmation de distribution.

## **ROCOD**

Rapports de confirmation de livraison requis.

Ce type de rapport est généré par le gestionnaire de files d'attente lorsqu'une application extrait le message de la file d'attente de destination d'une manière qui entraîne la suppression du message de la file d'attente. Les données du message d'origine ne sont pas incluses dans le message de rapport.

Si le message est extrait dans le cadre d'une unité de travail, le message de rapport est généré dans la même unité de travail, de sorte que le rapport n'est pas disponible tant que l'unité de travail n'est pas validée. Si l'unité de travail est annulée, le rapport n'est pas envoyé.

Aucun rapport COD n'est généré si la zone MDFMT du descripteur de message est FMDLH. Cela empêche la génération d'un rapport COD si le message n'est pas livrable et qu'il est placé dans une file d'attente de rebut.

ROCOD n'est pas valide si la file d'attente de destination est une file d'attente XCF.

N'indiquez pas plus d'un ROCOD, ROCODD ou ROCODF.

## **ROCODD**

Confirmez les rapports de livraison avec les données requises.

Cette valeur est identique à ROCOD, sauf que les 100 premiers octets des données de message d'application du message d'origine sont inclus dans le message de rapport. Si le message d'origine contient une ou plusieurs structures d'en-tête MQ, elles sont incluses dans le message de rapport, en plus des 100 octets de données d'application.

Si GMATM est spécifié dans l'appel MQGET pour le message d'origine et que le message extrait est tronqué, la quantité de données de message d'application placée dans le message de rapport est au minimum de:

- Longueur du message d'origine
- 100 octets.

ROCODD n'est pas valide si la file d'attente de destination est une file d'attente XCF.

N'indiquez pas plus d'un ROCOD, ROCODD ou ROCODF.

## **ROCODF**

Confirmez les rapports de livraison avec les données complètes requises.

Cette valeur est identique à ROCOD, sauf que toutes les données de message d'application du message d'origine sont incluses dans le message de rapport.

ROCODF n'est pas valide si la file d'attente de destination est une file d'attente XCF.

N'indiquez pas plus d'un ROCOD, ROCODD ou ROCODF.

**Options de notification d'action:** Vous pouvez spécifier l'une des options suivantes ou les deux pour demander à l'application de réception d'envoyer un message de rapport d'action positive ou d'action négative.

## **ROPAN**

Rapports de notification d'action positive requis.

Ce type de rapport est généré par l'application qui extrait le message et agit en conséquence. Elle indique que l'action demandée dans le message a été exécutée avec succès. L'application qui génère le rapport détermine si des données doivent être incluses dans le rapport.

Outre le fait de transmettre cette demande à l'application qui extrait le message, le gestionnaire de files d'attente n'effectue aucune action en fonction de cette option. Il incombe à l'application d'extraction de générer le rapport le cas échéant.

## **RONAN**

Des rapports de notification d'action négative sont requis.

Ce type de rapport est généré par l'application qui extrait le message et agit en conséquence. Il indique que l'action demandée dans le message n'a pas été exécutée correctement. L'application qui génère le rapport détermine si des données doivent être incluses dans le rapport. Par exemple, il peut être souhaitable d'inclure des données indiquant pourquoi la requête n'a pas pu être exécutée.

Outre le fait de transmettre cette demande à l'application qui extrait le message, le gestionnaire de files d'attente n'effectue aucune action en fonction de cette option. Il incombe à l'application d'extraction de générer le rapport le cas échéant.

La détermination des conditions qui correspondent à une action positive et celles qui correspondent à une action négative relève de la responsabilité de l'application. Cependant, il est recommandé que si la demande n'a été que partiellement exécutée, un rapport NAN plutôt qu'un rapport PAN soit généré si demandé. Il est également recommandé que toutes les conditions possibles correspondent soit à une action positive, soit à une action négative, mais pas aux deux.

**Options d'identificateur de message:** Vous pouvez spécifier l'une des options suivantes pour contrôler la manière dont le MDMID du message de rapport (ou du message de réponse) doit être défini.

#### **RONMI**

Nouvel identificateur de message.

Il s'agit de l'action par défaut, qui indique que si un rapport ou une réponse est généré suite à ce message, un nouveau MDMID doit être généré pour le message de rapport ou de réponse.

#### **ROPMI**

Transmettez l'identificateur de message.

Si un rapport ou une réponse est généré à la suite de ce message, le MDMID de ce message doit être copié dans le MDMID du message de rapport ou de réponse.

Le MsgId d'un message de publication sera différent pour chaque abonné qui reçoit une copie de la publication et, par conséquent, le MsgId copié dans le message de rapport ou de réponse sera différent pour chaque abonné.

Si cette option n'est pas spécifiée, RONMI est pris par défaut.

**Options d'identificateur de corrélation:** vous pouvez spécifier l'une des options suivantes pour contrôler la manière dont le MDCID du message de rapport (ou du message de réponse) doit être défini.

#### **ROCMTC**

Copiez l'identificateur de message dans l'identificateur de corrélation.

Il s'agit de l'action par défaut qui indique que si un rapport ou une réponse est généré à la suite de ce message, le MDMID de ce message doit être copié dans le MDCID du message de rapport ou de réponse.

Le MsgId d'un message de publication sera différent pour chaque abonné qui reçoit une copie de la publication et, par conséquent, le MsgId copié dans le CorrelId du message de rapport ou de réponse sera différent pour chacun d'eux.

#### **ROPCI**

Transmettez l'identificateur de corrélation.

Si un rapport ou une réponse est généré à la suite de ce message, le MDCID de ce message doit être copié dans le MDCID du message de rapport ou de réponse.

Le MDCID d'un message de publication est spécifique à un abonné, sauf s'il utilise l'option SOSCID et définit la zone SCDIC dans le MQSD sur CINONE. Par conséquent, il est possible que le MDCID copié dans le MDCID du message de rapport ou de réponse soit différent pour chacun d'eux.

Si cette option n'est pas spécifiée, ROCMTC est utilisé.

Les serveurs qui répondent aux demandes ou qui génèrent des messages de rapport sont recommandés pour vérifier si les options ROPMI ou ROPCI ont été définies dans le message d'origine.

Si tel est le cas, les serveurs doivent effectuer l'action décrite pour ces options. Si aucun de ces deux paramètres n'est défini, les serveurs doivent effectuer l'action par défaut correspondante.

: Vous pouvez spécifier l'une des options suivantes pour contrôler la disposition du message d'origine lorsqu'il ne peut pas être distribué à la file d'attente de destination. Ces options s'appliquent uniquement aux situations qui entraînent la génération d'un message de rapport d'exception si l'application émettrice en a demandé un. L'application peut définir les options d'élimination indépendamment de la demande de rapports d'exception.

### **RODLQ**

Placez le message dans la file d'attente des messages non livrés.

Il s'agit de l'action par défaut, qui indique que le message doit être placé dans la file d'attente de rebut, si le message ne peut pas être distribué à la file d'attente de destination. Cela se produit dans les situations suivantes:

- Lorsque l'application qui a inséré le message d'origine ne peut pas être informée de manière synchrone du problème à l'aide du code anomalie renvoyé par l'appel MQPUT ou MQPUT1 . Un message de rapport d'exception est généré si l'expéditeur en a demandé un.
- Lorsque l'application qui a inséré le message d'origine a inséré une rubrique

Un message de rapport d'exception est généré si l'expéditeur en a demandé un.

### **RODISC**

Supprimer le message.

Cela indique que le message doit être supprimé s'il ne peut pas être distribué à la file d'attente de destination. Cela se produit dans les situations suivantes:

- Lorsque l'application qui a inséré le message d'origine ne peut pas être informée de manière synchrone du problème à l'aide du code anomalie renvoyé par l'appel MQPUT ou MQPUT1 . Un message de rapport d'exception est généré si l'expéditeur en a demandé un.
- Lorsque l'application qui a inséré le message d'origine a inséré une rubrique

Un message de rapport d'exception est généré si l'expéditeur en a demandé un.

S'il est nécessaire de renvoyer le message d'origine à l'expéditeur, sans que le message d'origine soit placé dans la file d'attente des messages non livrés, l'expéditeur doit spécifier RODISC avec ROEXCF.

**Option par défaut:** Vous pouvez spécifier ce qui suit si aucune option de rapport n'est requise:

### **RONONE**

Aucun rapport n'est requis.

Cette valeur peut être utilisée pour indiquer qu'aucune autre option n'a été spécifiée. RONONE est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette option soit utilisée avec une autre option, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

### **Informations générales :**

1. Tous les types de rapport requis doivent être spécifiquement demandés par l'application qui envoie le message d'origine. Par exemple, si un rapport COA est demandé mais qu'un rapport d'exception ne l'est pas, un rapport COA est généré lorsque le message est placé dans la file d'attente de destination, mais aucun rapport d'exception n'est généré si la file d'attente de destination est saturée lorsque le message y arrive. Si aucune option MDREP n'est définie, aucun message de rapport n'est généré par le gestionnaire de files d'attente ou l'agent MCA.

Certaines options de rapport peuvent être spécifiées même si le gestionnaire de files d'attente local ne les reconnaît pas ; cela est utile lorsque l'option doit être traitée par le gestionnaire de files d'attente de destination. Voir [«Options de rapport et indicateurs de message sous IBM i»](#), à la page 1505 pour plus de détails.

Si un message de rapport est demandé, le nom de la file d'attente à laquelle le rapport doit être envoyé doit être spécifié dans la zone MDRQ . Lorsqu'un message de rapport est reçu, la nature du rapport peut être déterminée en examinant la zone MDFB dans le descripteur de message.

2. Si le gestionnaire de files d'attente ou l'agent MCA qui génère un message de rapport ne parvient pas à placer le message de rapport dans la file d'attente de réponses (par exemple, parce que la file d'attente de réponses ou la file d'attente de transmission est saturée), le message de rapport est placé dans la file d'attente de rebut. Si cela échoue également ou s'il n'existe pas de file d'attente de rebut, l'action effectuée dépend du type de message de rapport:
  - Si le message de rapport est un rapport d'exception, le message qui a généré le rapport d'exception est laissé dans sa file d'attente de transmission, ce qui garantit que le message n'est pas perdu.
  - Pour tous les autres types de rapport, le message de rapport est supprimé et le traitement se poursuit normalement. Cela est dû au fait que le message d'origine a déjà été distribué en toute sécurité (pour les messages de rapport COA ou COD) ou qu'il n'a plus d'intérêt (pour un message de rapport d'expiration).

Une fois qu'un message de rapport a été correctement placé dans une file d'attente (file de destination ou file de transmission intermédiaire), le message n'est plus soumis à un traitement spécial ; il est traité comme tout autre message.

3. Lorsque le rapport est généré, la file d'attente MDRQ est ouverte et le message de rapport est inséré à l'aide des droits de l'utilisateur MDUID dans le MQMD du message à l'origine du rapport, sauf dans les cas suivants:
  - Les rapports d'exception générés par un agent MCA récepteur sont placés avec les droits utilisés par l'agent MCA lorsqu'il a tenté d'insérer le message à l'origine du rapport. L'attribut de canal CDPA détermine l'identificateur utilisateur utilisé.
  - Les rapports COA générés par le gestionnaire de files d'attente sont insérés avec les droits d'accès utilisés lorsque le message à l'origine du rapport a été inséré dans le gestionnaire de files d'attente qui génère le rapport. Par exemple, si le message a été inséré par un agent MCA récepteur à l'aide de l'identificateur utilisateur de l'agent MCA, le gestionnaire de files d'attente insère le rapport COA à l'aide de l'identificateur utilisateur de l'agent MCA.

Les applications qui génèrent des rapports doivent normalement utiliser les mêmes droits que ceux qu'elles auraient utilisés pour générer une réponse ; il s'agit normalement des droits de l'ID utilisateur dans le message d'origine.

Si le rapport doit être acheminé vers une destination éloignée, les expéditeurs et les destinataires peuvent décider de l'accepter ou non, de la même manière qu'ils le font pour les autres messages.

4. Si un message de rapport contenant des données est demandé:
  - Le message de rapport est toujours généré avec la quantité de données demandée par l'expéditeur du message d'origine. Si le message d'état est trop volumineux pour la file d'attente de réponses, le traitement décrit précédemment se produit ; le message d'état n'est jamais tronqué pour tenir dans la file d'attente de réponses.
  - Si le MDFMT du message d'origine est FMXQH, les données incluses dans le rapport n'incluent pas le MQXQH. Les données de rapport commencent par le premier octet des données au-delà du MQXQH dans le message d'origine. Cela se produit si la file d'attente est une file d'attente de transmission.
5. Si un message COA, COD ou de rapport d'expiration est reçu dans la file d'attente de réponses, il est garanti que le message d'origine est arrivé, a été distribué ou a expiré, selon le cas. Toutefois, si un ou plusieurs de ces messages de rapport sont demandés et ne sont pas reçus, l'inverse ne peut pas être pris en compte, car l'un des cas suivants peut s'être produit:
  - a. Le message de rapport est suspendu car un lien est arrêté.
  - b. Le message de rapport est suspendu car une condition de blocage existe dans une file d'attente de transmission intermédiaire ou dans la file d'attente de réponse (par exemple, la file d'attente est saturée ou bloquée pour les insertions).



- c. Le message de rapport se trouve dans une file d'attente de rebut.
- d. Lorsque le gestionnaire de files d'attente a tenté de générer le message de rapport, il n'a pas pu le placer dans la file d'attente appropriée et il n'a pas non plus pu le placer dans la file d'attente de rebut, de sorte que le message de rapport n'a pas pu être généré.
- e. Une défaillance du gestionnaire de files d'attente s'est produite entre l'action signalée (arrivée, distribution ou expiration) et la génération du message de rapport correspondant. (Cela ne se produit pas pour les messages de rapport COD si l'application extrait le message d'origine dans une unité de travail, car le message de rapport COD est généré dans la même unité de travail.)

Les messages de rapport d'exception peuvent être suspendus de la même manière pour les raisons 1, 2 et 3 précédemment. Toutefois, lorsqu'un agent MCA ne parvient pas à générer un message de rapport d'exception (le message de rapport ne peut pas être placé dans la file d'attente de réponse ou dans la file d'attente de rebut), le message d'origine reste dans la file d'attente de transmission de l'émetteur et le canal est fermé. Cela se produit indépendamment du fait que le message de rapport devait être généré à l'extrémité émettrice ou réceptrice du canal.

6. Si le message d'origine est temporairement bloqué (ce qui entraîne la génération d'un message de rapport d'exception et l'insertion du message d'origine dans une file d'attente de messages non livrés), mais que le blocage est effacé et qu'une application lit ensuite le message d'origine dans la file d'attente de messages non livrés et le place à nouveau à sa destination, les situations suivantes peuvent se produire:
  - Même si un message de rapport d'exception a été généré, le message d'origine arrive finalement à sa destination.
  - Plusieurs messages de rapport d'exception sont générés pour un seul message d'origine, car le message d'origine peut rencontrer un autre blocage ultérieurement.

#### **Signaler les messages lors de l'insertion dans une rubrique:**

1. Des rapports peuvent être générés lors de l'insertion d'un message dans une rubrique. Ce message sera envoyé à tous les abonnés à la rubrique, qui peut être zéro, un ou plusieurs. Cela doit être pris en compte lorsque vous choisissez d'utiliser les options de rapport car un grand nombre de messages de rapport peuvent être générés en conséquence.
2. Lors de l'insertion d'un message dans une rubrique, il se peut que de nombreuses files d'attente de destination reçoivent une copie du message. Si certaines de ces files d'attente de destination présentent un problème, par exemple une file d'attente saturée, la réussite de l'opération MQPUT dépend de la valeur de NPMMSGDLV ou de PMSGDLV (en fonction de la persistance du message). Si le paramètre est tel que la distribution des messages à la file d'attente de destination doit aboutir (par exemple, s'il s'agit d'un message persistant envoyé à un abonné durable et que PMSGDLV est défini sur ALL ou ALLDUR), la réussite est définie comme l'un des critères suivants:
  - Insertion réussie dans la file d'attente de l'abonné
  - Utilisation de RODLQ et insertion réussie dans la file d'attente de rebut si la file d'attente de l'abonné ne peut pas prendre le message
  - Utilisation de RODISC si la file d'attente de l'abonné ne peut pas prendre le message.

#### **Messages de rapport pour les segments de message:**

1. Des messages de rapport peuvent être demandés pour les messages dont la segmentation est autorisée (voir la description de l'indicateur MFSEGA). Si le gestionnaire de files d'attente juge nécessaire de segmenter le message, un message de rapport peut être généré pour chacun des segments qui rencontre ensuite la condition appropriée. Les applications doivent donc être préparées pour recevoir plusieurs messages de rapport pour chaque type de message de rapport demandé. La zone MDGID du message de rapport peut être utilisée pour corréler les différents rapports avec l'identificateur de groupe du message d'origine et la zone MDFB pour identifier le type de chaque message de rapport.
2. Si GMLOGO est utilisé pour extraire des messages de rapport pour des segments, sachez que des rapports de types différents peuvent être renvoyés par les appels MQGET successifs. Par exemple, si les rapports COA et COD sont demandés pour un message segmenté par le gestionnaire de files

d'attente, les appels MQGET des messages de rapport peuvent renvoyer les messages de rapport COA et COD imbriqués de manière imprévisible. Ceci peut être évité en utilisant l'option GMCMPM (éventuellement avec GMATM). GMCMPM permet au gestionnaire de files d'attente de réassembler des messages de rapport ayant le même type de rapport. Par exemple, le premier appel MQGET peut réassembler tous les messages COA relatifs au message d'origine et le deuxième appel MQGET peut réassembler tous les messages COD. Ce qui est réassemblé en premier dépend du type de message de rapport qui apparaît en premier dans la file d'attente.

3. Les applications qui elles-mêmes placent des segments peuvent spécifier des options de rapport différentes pour chaque segment. Toutefois, il convient de noter les points suivants:
  - Si les segments sont extraits à l'aide de l'option GMCMPM, seules les options de rapport du premier segment sont prises en compte par le gestionnaire de files d'attente.
  - Si les segments sont extraits un par un, et que la plupart d'entre eux ont l'une des options ROCOD\*, mais qu'au moins un segment ne le fait pas, il ne sera pas possible d'utiliser l'option GMCMPM pour extraire les messages de rapport avec un seul appel MQGET, ou d'utiliser l'option GMASGA pour détecter quand tous les messages de rapport sont arrivés.
4. Dans un réseau MQ, il est possible que les gestionnaires de files d'attente aient des capacités différentes. Si un message de rapport pour un segment est généré par un gestionnaire de files d'attente ou un agent MCA qui ne prend pas en charge la segmentation, le gestionnaire de files d'attente ou l'agent MCA n'inclut pas par défaut les informations de segment nécessaires dans le message de rapport, ce qui peut rendre difficile l'identification du message d'origine à l'origine de la génération du rapport. Cette difficulté peut être évitée en demandant des données avec le message de rapport, c'est-à-dire en spécifiant les options RO\* D ou RO\* F appropriées. Toutefois, sachez que si RO\* D est spécifié, moins de 100 octets de données de message d'application peuvent être renvoyés à l'application qui extrait le message de rapport, si le message de rapport est généré par un gestionnaire de files d'attente ou un agent MCA qui ne prend pas en charge la segmentation.

**Contenu du descripteur de message d'un message de rapport:** Lorsque le gestionnaire de files d'attente ou l'agent MCA génère un message de rapport, il définit les zones du descripteur de message sur les valeurs suivantes, puis place le message de manière normale.

Tableau 708. Valeurs utilisées pour les zones MQMD lorsqu'un message de rapport est généré par le système

Zone dans MQMD	Valeur utilisée
MDSID	MSIDV (MDSID)
MDVER	MDVER2
MDREP	RONONE
MDMT	MTRPRT
MDEXP	EIULIM
MDFB	Selon la nature du rapport (FBCOA, FBCOD, FBEXP ou une valeur RC*)
MDENC	Copié à partir du descripteur de message d'origine
MDCSI	Copié à partir du descripteur de message d'origine
MDFMT	Copié à partir du descripteur de message d'origine
MDPRI	Copié à partir du descripteur de message d'origine
MDPER	Copié à partir du descripteur de message d'origine
MDMID	Comme indiqué par les options de rapport dans le descripteur de message d'origine
MDCID	Comme indiqué par les options de rapport dans le descripteur de message d'origine
MDBOC	0

Tableau 708. Valeurs utilisées pour les zones MQMD lorsqu'un message de rapport est généré par le système (suite)

Zone dans MQMD	Valeur utilisée
MDRQ	Espaces vides
MDRM	Nom du gestionnaire de files d'attente
MDUID	Comme défini par l'option PMPASI
MDACC	Comme défini par l'option PMPASI
MDAID	Comme défini par l'option PMPASI
MDPAT	ATQM, ou selon le cas, pour l'agent MCA
MDPAN	28 premiers octets du nom du gestionnaire de files d'attente ou du nom de l'agent MCA. Pour les messages de rapport générés par le pont IMS, cette zone contient le nom de groupe XCF et le nom de membre XCF du système IMS auquel le message est lié.
MDPD	Date d'envoi du message de rapport
MDPT	Heure d'envoi du message de rapport
MDAOD	Espaces vides
MDGID	Copié à partir du descripteur de message d'origine
MDSEQ	Copié à partir du descripteur de message d'origine
MDOFF	Copié à partir du descripteur de message d'origine
MDMFL	Copié à partir du descripteur de message d'origine
MDOLN	Copié à partir du descripteur de message d'origine s'il ne s'agit pas d'OLUNDF, et défini sur la longueur des données de message d'origine dans le cas contraire

Une application générant un rapport est recommandée pour définir des valeurs similaires, à l'exception des suivantes:

- La zone MDRM peut être mise à blanc (le gestionnaire de files d'attente change le nom du gestionnaire de files d'attente local lorsque le message est inséré).
- Les zones de contexte doivent être définies à l'aide de l'option qui aurait été utilisée pour une réponse, normalement PMPASI.

**Analyse de la zone de rapport:** la zone MDREP contient des sous-zones ; pour cette raison, les applications qui doivent vérifier si l'expéditeur du message a demandé un rapport particulier doivent utiliser l'une des techniques décrites dans [«Analyse de la zone de rapport sur IBM i»](#), à la page 1507.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1. La valeur initiale de cette zone est RONONE.

#### **MDRM (chaîne de caractères de 48 octets)**

Nom du gestionnaire de files d'attente de réponses.

Il s'agit du nom du gestionnaire de files d'attente auquel le message de réponse ou de rapport doit être envoyé. MDRQ est le nom local d'une file d'attente définie sur ce gestionnaire de files d'attente.

Si la zone MDRM est vide, le gestionnaire de files d'attente local recherche le nom **MDRQ** dans ses définitions de file d'attente. S'il existe une définition locale d'une file d'attente éloignée portant ce nom, la valeur **MDRM** du message transmis est remplacée par la valeur de l'attribut **RemoteQMgrName** de la définition de la file d'attente éloignée. Cette valeur est renvoyée dans le descripteur de message lorsque l'application de réception émet un appel MQGET pour le message. S'il n'existe pas de

définition locale d'une file d'attente éloignée, le MDRM transmis avec le message est le nom du gestionnaire de files d'attente local.

Si le nom est indiqué, il peut contenir des blancs de fin ; le premier caractère nul et les caractères qui le suivent sont traités comme des blancs. Dans le cas contraire, il n'est pas vérifié que le nom respecte les règles de dénomination des gestionnaires de files d'attente ou que ce nom est connu du gestionnaire de files d'attente émetteur ; cela est également vrai pour le nom transmis, si **MDRM** est remplacé dans le message transmis.

Si une file d'attente de réponse n'est pas requise, il est recommandé (même si cette case n'est pas cochée) que la zone MDRM soit mise à blanc ; la zone ne doit pas être laissée non initialisée.

Pour l'appel MQGET, le gestionnaire de files d'attente renvoie toujours le nom complété par des blancs à la longueur de la zone.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La longueur de cette zone est indiquée par LNQM. La valeur initiale de cette zone est de 48 caractères blancs.

### **MDRQ (chaîne de caractères de 48 octets)**

Nom de la file d'attente de réponses.

Il s'agit du nom de la file d'attente de messages à laquelle l'application qui a émis la demande d'obtention du message doit envoyer les messages MTRPLY et MTRPRT. Il s'agit du nom local d'une file d'attente définie sur le gestionnaire de files d'attente identifié par MDRM. Cette file d'attente ne doit pas être une file d'attente modèle, bien que le gestionnaire de files d'attente émetteur ne la vérifie pas lorsque le message est inséré.

Pour les appels MQPUT et MQPUT1 , cette zone ne doit pas être vide si la zone MDMT a la valeur MTRQST ou si des messages de rapport sont demandés par la zone MDREP . Toutefois, la valeur spécifiée (ou remplacée) est transmise à l'application qui émet la demande d'obtention du message, quel que soit le type de message.

Si la zone MDRM est vide, le gestionnaire de files d'attente local recherche le nom MDRQ dans ses propres définitions de file d'attente. S'il existe une définition locale d'une file d'attente éloignée portant ce nom, la valeur MDRQ du message transmis est remplacée par la valeur de l'attribut **RemoteQName** de la définition de la file d'attente éloignée. Cette valeur est renvoyée dans le descripteur de message lorsque l'application de réception émet un appel MQGET pour le message. Si une définition locale d'une file d'attente éloignée n'existe pas, MDRQ n'est pas modifié.

Si le nom est indiqué, il peut contenir des blancs de fin ; le premier caractère nul et les caractères qui le suivent sont traités comme des blancs. Dans le cas contraire, il n'est pas vérifié que le nom respecte les règles de dénomination des files d'attente ; cela est également vrai pour le nom transmis, si MDRQ est remplacé dans le message transmis. La seule vérification effectuée est qu'un nom a été spécifié, si les circonstances l'exigent.

Si une file d'attente de réponse n'est pas requise, il est recommandé (même si cette case n'est pas cochée) que la zone MDRQ soit mise à blanc ; la zone ne doit pas être laissée non initialisée.

Pour l'appel MQGET, le gestionnaire de files d'attente renvoie toujours le nom complété par des blancs à la longueur de la zone.

Si un message nécessitant un message de rapport ne peut pas être distribué et que le message de rapport ne peut pas non plus être distribué dans la file d'attente spécifiée, le message d'origine et le message de rapport sont placés dans la file d'attente de rebut (message non distribué). Voir l'attribut **DeadLetterQName** décrit dans [«Attributs du gestionnaire de files d'attente sous IBM i»](#), à la page 1469.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La longueur de cette zone est indiquée par LNQN. La valeur initiale de cette zone est de 48 caractères blancs.

### **MDSEQ (entier signé à 10 chiffres)**

Numéro de séquence du message logique dans le groupe.

Les numéros de séquence commencent à 1 et augmentent de 1 pour chaque nouveau message logique du groupe, jusqu'à un maximum de 999 999 999. Un message physique qui ne fait pas partie d'un groupe a le numéro de séquence 1.

Cette zone n'a pas besoin d'être définie par l'application sur l'appel MQPUT ou MQGET si:

- Dans l'appel MQPUT, PMLOGO est spécifié.
- Dans l'appel MQGET, MOSEQN n'est pas spécifié.

Il s'agit des méthodes recommandées pour utiliser ces appels pour les messages qui ne sont pas des messages de rapport. Toutefois, si l'application requiert davantage de contrôle ou si l'appel est MQPUT1, l'application doit s'assurer que MDSEQ est défini sur une valeur appropriée.

Lors de l'entrée dans les appels MQPUT et MQPUT1, le gestionnaire de files d'attente utilise la valeur détaillée dans le [Tableau 1](#). Dans la sortie des appels MQPUT et MQPUT1, le gestionnaire de files d'attente définit cette zone sur la valeur qui a été envoyée avec le message.

En entrée de l'appel MQGET, le gestionnaire de files d'attente utilise la valeur détaillée dans le [Tableau 1](#). Dans la sortie de l'appel MQGET, le gestionnaire de files d'attente définit cette zone sur la valeur du message extrait.

La valeur initiale de cette zone est 1. Cette zone est ignorée si MDVER est inférieur à MDVER2.

### **MDSID (chaîne de caractères de 4 octets)**

Identificateur de structure.

La valeur doit être:

#### **MSIDV (MDSID)**

Identificateur de la structure de descripteur de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MDSIDV.

### **MDUID (chaîne de caractères de 12 octets)**

ID utilisateur.


Il fait partie du *contexte d'identité* du message. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

MDUID indique l'ID utilisateur de l'application à l'origine du message. Le gestionnaire de files d'attente traite ces informations comme des données de type caractères, mais ne définit pas leur format.

Une fois qu'un message a été reçu, MDUID peut être utilisé dans la zone ODAU du paramètre **OBJDSC** d'un appel MQOPEN ou MQPUT1 ultérieur, de sorte que la vérification d'autorisation soit effectuée pour l'utilisateur MDUID au lieu de l'application effectuant l'ouverture.

Lorsque le gestionnaire de files d'attente génère ces informations pour un appel MQPUT ou MQPUT1, il utilise un identificateur utilisateur déterminé à partir de l'environnement.

Lorsque l'ID utilisateur est déterminé à partir de l'environnement:

-  Sous z/OS, le gestionnaire de files d'attente utilise:
  - Pour le traitement par lots, l'ID utilisateur de la carte de travail JES ou de la tâche démarrée
  - Pour TSO, l'ID utilisateur de connexion
  - Pour CICS, l'identificateur utilisateur associé à la tâche
  - Pour IMS, l'identificateur d'utilisateur dépend du type d'application:
    - Pour :
      - Régions BMP sans message
      - Régions IFP sans message
      - Régions BMP de message et IFP de message qui n'ont pas émis d'appel GU réussi




le gestionnaire de files d'attente utilise l'ID utilisateur de la carte JOB JES de la région ou l'ID utilisateur TSO. S'ils sont vides ou nuls, ils utilisent le nom du bloc de spécification de programme (PSB).

- Pour :

- Régions BMP de message et IFP de message ayant émis un appel GU réussi
- Régions MPP

le gestionnaire de files d'attente utilise l'un des éléments suivants:

- ID utilisateur connecté associé au message
- Nom du terminal logique (LTERM)
- Identificateur utilisateur de la carte de travail JES de la région
- ID utilisateur TSO
- Nom du bloc de spécification de programme

-  Sous IBM i, le gestionnaire de files d'attente utilise le nom du profil utilisateur associé au travail d'application.
-  Sous AIX and Linux, le gestionnaire de files d'attente utilise:
  - Nom de connexion de l'application
  - ID utilisateur effectif du processus si aucune connexion n'est disponible
  - Identificateur utilisateur associé à la transaction, si l'application est une transaction CICS
- Sous VSE/ESA, il s'agit d'une zone réservée.
-  Sous Windows, le gestionnaire de files d'attente utilise les 12 premiers caractères du nom d'utilisateur connecté.

Pour les appels MQPUT et MQPUT1, il s'agit d'une zone d'entrée-sortie si PMSETI ou PMSETA est spécifié dans le paramètre **PMO**. Toutes les informations qui suivent un caractère null dans la zone sont supprimées. Le caractère null et tous les caractères suivants sont convertis en blancs par le gestionnaire de files d'attente. Si PMSETI ou PMSETA n'est pas indiqué, cette zone est ignorée en entrée et est une zone de sortie uniquement.

Une fois qu'un appel MQPUT ou MQPUT1 a abouti, cette zone contient le MDUID qui a été transmis avec le message s'il a été inséré dans une file d'attente. Il s'agit de la valeur de MDUID qui est conservée avec le message s'il est conservé (voir la description de PMRET pour plus de détails sur les publications conservées) mais qui n'est pas utilisée en tant que MDUID lorsque le message est envoyé en tant que publication aux abonnés car ils fournissent une valeur pour remplacer MDUID dans toutes les publications qui leur sont envoyées. Si le message n'a pas de contexte, la zone est entièrement vide.

Il s'agit d'une zone de sortie pour l'appel MQGET. La longueur de cette zone est indiquée par LNUID. La valeur initiale de cette zone est de 12 caractères blancs.

### **MDVER (entier signé à 10 chiffres)**

Numéro de version de la structure.

Il doit s'agir de l'une des valeurs suivantes :

#### **MDVER1**

Structure de descripteur de message Version-1 .

#### **MDVER2**

Structure de descripteur de message Version-2 .

**Remarque :** Lorsqu'un MQMD version-2 est utilisé, le gestionnaire de files d'attente effectue des vérifications supplémentaires sur les structures d'en-tête MQ qui peuvent être présentes au début

des données de message d'application. Pour plus de détails, voir les remarques sur l'utilisation de l'appel MQPUT.

Les zones qui existent uniquement dans la version la plus récente de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

#### **MDVERC**

Version actuelle de la structure de descripteur de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MDVER1.

### **Valeurs initiales**

*Tableau 709. Zones dans MQMD*

<b>Nom de zone</b>	<b>Nom de la constante</b>	<b>Valeur de la constante</b>
MDSID	MSIDV (MDSID)	'MD--'
MDVER	MDVER1	1
MDREP	RONONE	0
MDMT	MTDGRM	8
MDEXP	EIULIM	-1
MDFB	FBNONE	0
MDENC	ENNAT	Dépend de l'environnement
MDCSI	CQM	0
MDFMT	FMAUCUN	Espaces vides
MDPRI	PRQDEF	-1
MDPER	PEQDEF	2
MDMID	MINONE	Valeurs NULL
MDCID	CINONE	Valeurs NULL
MDBOC	Aucun	0
MDRQ	Aucun	Espaces vides
MDRM	Aucun	Espaces vides
MDUID	Aucun	Espaces vides
MDACC	ACNONE	Valeurs NULL
MDAID	Aucun	Espaces vides
MDPAT	ATNCON	0
MDPAN	Aucun	Espaces vides
MDPD	Aucun	Espaces vides
MDPT	Aucun	Espaces vides
MDAOD	Aucun	Espaces vides
MDGID	GINONE	Valeurs NULL
MDSEQ	Aucun	1

Tableau 709. Zones dans MQMD (suite)

Nom de zone	Nom de la constante	Valeur de la constante
MDOFF	Aucun	0
MDMFL	MFNONE	0
MDOLN	ANNULER	-1

**Remarques :**

- Le symbole ~ représente un caractère blanc unique.

## Déclaration RPG

```

D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQMD Structure
D*
D* Structure identifier
D MDSID          1      4  INZ('MD ')
D* Structure version number
D MDVER          5      8I 0 INZ(1)
D* Options for report messages
D MDREP          9     12I 0 INZ(0)
D* Message type
D MDMT          13     16I 0 INZ(8)
D* Message lifetime
D MDEXP         17     20I 0 INZ(-1)
D* Feedback or reason code
D MDFB          21     24I 0 INZ(0)
D* Numeric encoding of message data
D MDENC         25     28I 0 INZ(273)
D* Character set identifier of messagedata
D MDCSI         29     32I 0 INZ(0)
D* Format name of message data
D MDFMT         33     40  INZ(' ')
D* Message priority
D MDPRI         41     44I 0 INZ(-1)
D* Message persistence
D MDPER         45     48I 0 INZ(2)
D* Message identifier
D MDMID         49     72  INZ(X'00000000000000-
D                    00000000000000000000-
D                    000000000000')
D* Correlation identifier
D MDCID         73     96  INZ(X'00000000000000-
D                    00000000000000000000-
D                    000000000000')
D* Backout counter
D MDBOC         97    100I 0 INZ(0)
D* Name of reply queue
D MDRQ         101    148  INZ
D* Name of reply queue manager
D MDRM         149    196  INZ
D* User identifier
D MDUID        197    208  INZ
D* Accounting token
D MDACC        209    240  INZ(X'00000000000000-
D                    00000000000000000000-
D                    00000000000000000000-
D                    000000')
D* Application data relating to identity
D MDAID        241    272  INZ
D* Type of application that put the message
D MDPAT        273    276I 0 INZ(0)
D* Name of application that put the message
D MDPAN        277    304  INZ
D* Date when message was put
D MDPD        305    312  INZ
D* Time when message was put
D MDPT        313    320  INZ
D* Application data relating to origin
D MDAOD        321    324  INZ
D* Group identifier

```



```

D  MDGID          325    348    INZ(X'0000000000000000-
D                    000000000000000000000000-
D                    000000000000')
D* Sequence number of logical message within group
D  MDSEQ          349    352I 0 INZ(1)
D* Offset of data in physical message from start of logical message
D  MDOFF          353    356I 0 INZ(0)
D* Message flags
D  MDMFL          357    360I 0 INZ(0)
D* Length of original message
D  MDOLN          361    364I 0 INZ(-1)

```

IBM i

## MQMDE (extension de descripteur de message) sous IBM i

### Présentation

**Objectif:** La structure MQMDE décrit les données qui précèdent parfois les données de message d'application. La structure contient les zones MQMD qui existent dans le MQMD version-2, mais pas dans le MQMD version-1.

**Nom de format:** FMMDE.

**Jeu de caractères et codage:** les données de MQMDE doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par ENNAT pour le langage de programmation C.

Le jeu de caractères et le codage de MQMDE doivent être définis dans les zones *MDCSI* et *MDENC* dans:

- MQMD (si la structure MQMDE est au début des données de message), ou
- Structure d'en-tête qui précède la structure MQMDE (tous les autres cas).

Si le MQMDE ne fait pas partie du jeu de caractères et du codage du gestionnaire de files d'attente, il est accepté mais non respecté, c'est-à-dire que le MQMDE est traité comme des données de message.

**Utilisation:** les applications normales doivent utiliser un MQMD version-2, auquel cas elles ne seront pas confrontées à une structure MQMDE. Toutefois, les applications spécialisées et les applications qui continuent d'utiliser un MQMD version-1 peuvent rencontrer un MQMDE dans certaines situations. La structure MQMDE peut se produire dans les cas suivants:

- Spécifié sur les appels MQPUT et MQPUT1
- Renvoyé par l'appel MQGET
- Dans les messages des files d'attente de transmission
- «MQMDE spécifié sur les appels MQPUT et MQPUT1», à la page [1209](#)
- «MQMDE renvoyé par l'appel MQGET», à la page [1210](#)
- «MQMDE dans les messages des files d'attente de transmission», à la page [1210](#)
- «Zones», à la page [1211](#)
- «Valeurs initiales», à la page [1213](#)
- «Déclaration RPG», à la page [1213](#)

### MQMDE spécifié sur les appels MQPUT et MQPUT1

Dans les appels MQPUT et MQPUT1, si l'application fournit un MQMD version-1, l'application peut éventuellement préfixer les données de message avec un MQMDE, en définissant la zone *MDFMT* dans MQMD sur FMMDE pour indiquer qu'un MQMDE est présent. Si l'application ne fournit pas de MQMDE, le gestionnaire de files d'attente utilise les valeurs par défaut pour les zones du MQMDE. Les valeurs par défaut utilisées par le gestionnaire de files d'attente sont les mêmes que les valeurs initiales de la structure (voir [Tableau 711](#), à la page [1213](#)).

Si l'application fournit un version-2 MQMD et préfixe les données de message d'application avec un MQMDE, les structures sont traitées comme indiqué dans la [Tableau 710](#), à la page [1210](#).

Tableau 710. Action du gestionnaire de files d'attente lorsque MQMDE est spécifié sur MQPUT ou MQPUT1

Version MQMD	Valeurs des zones version-2	Valeurs des zones correspondantes dans MQMDE	Action effectuée par le gestionnaire de files d'attente
1	-	Valide	MQMDE est honoré
2	Par défaut	Valide	MQMDE est honoré
2	Pas de valeur par défaut	Valide	MQMDE est traité comme des données de message
1 ou 2	Tous	Non valide	L'appel échoue avec un code anomalie approprié
1 ou 2	Tous	MQMDE est dans un jeu de caractères ou un codage incorrect, ou est une version non prise en charge	MQMDE est traité comme des données de message

Il y a un cas particulier. Si l'application utilise un MQMD version-2 pour insérer un message qui est un segment (c'est-à-dire que l'indicateur MFSEG ou MFLSEG est défini) et que le nom de format dans le MQMD est FMDLH, le gestionnaire de files d'attente génère une structure MQMDE et l'insère *entre* la structure MQDLH et les données qui la suivent. Dans le MQMD conservé par le gestionnaire de files d'attente avec le message, les zones version-2 sont définies sur leurs valeurs par défaut.

Plusieurs des zones qui existent dans le MQMD version-2 mais pas dans le MQMD version-1 sont des zones d'entrée-sortie dans MQPUT et MQPUT1. Toutefois, le gestionnaire de files d'attente ne renvoie aucune valeur dans les zones équivalentes de MQMDE dans la sortie des appels MQPUT et MQPUT1 ; si l'application requiert ces valeurs de sortie, elle doit utiliser un MQMD version-2 .

## MQMDE renvoyé par l'appel MQGET

Dans l'appel MQGET, si l'application fournit un MQMD version-1 , le gestionnaire de files d'attente préfixe le message renvoyé avec un MQMDE, mais uniquement si une ou plusieurs des zones du MQMDE ont une valeur autre que la valeur par défaut. Le gestionnaire de files d'attente définit la zone *MDFMT* dans MQMD sur la valeur FMMDE pour indiquer qu'un MQMDE est présent.

Si l'application fournit un MQMDE au début du paramètre **BUFFER** , MQMDE est ignoré. En retour de l'appel MQGET, il est remplacé par le MQMDE pour le message (si nécessaire) ou remplacé par les données de message d'application (si le MQMDE n'est pas nécessaire).

Si un MQMDE est renvoyé par l'appel MQGET, les données du MQMDE se trouvent généralement dans le jeu de caractères et le codage du gestionnaire de files d'attente. Cependant, le MQMDE peut être dans un autre jeu de caractères et codage si:

- Le MQMDE a été traité comme des données sur l'appel MQPUT ou MQPUT1 (voir [Tableau 710](#), à la page [1210](#) pour connaître les circonstances qui peuvent être à l'origine de cette erreur).
- Le message a été reçu d'un gestionnaire de files d'attente éloignées connecté par une connexion TCP et l'agent MCA (Message Channel Agent) récepteur n'a pas été configuré correctement (voir [Sécurité des objets IBM MQ for IBM i](#) pour plus d'informations).

## MQMDE dans les messages des files d'attente de transmission

Les messages des files d'attente de transmission sont préfixés avec la structure MQXQH, qui contient un MQMD version-1 . Un MQMDE peut également être présent, positionné entre la structure MQXQH et les données de message d'application, mais il ne sera généralement présent que si une ou plusieurs des zones du MQMDE ont une valeur autre que la valeur par défaut.

D'autres structures d'en-tête IBM MQ peuvent également se produire entre la structure MQXQH et les données de message d'application. Par exemple, lorsque l'en-tête de rebut MQDLH est présent et que le message n'est pas un segment, l'ordre est le suivant:

- MQXQH (contenant un MQMD version-1 )
- MQMDE
- MQDLH
- Données de message d'application

## Zones

La structure MQMDE contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

### **MECSI (entier signé à 10 chiffres)**

Identificateur de jeu de caractères des données qui suivent MQMDE.

Indique l'identificateur de jeu de caractères des données qui suivent la structure MQMDE ; il ne s'applique pas aux données de type caractères de la structure MQMDE elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Le gestionnaire de files d'attente ne vérifie pas que cette zone est valide. La valeur spéciale suivante peut être utilisée:

#### **CSINHT**

Hérite de l'identificateur de jeu de caractères de cette structure.

Les données de type caractères dans les données *suivant* cette structure se trouvent dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, la valeur CSINHT n'est pas renvoyée par l'appel MQGET.

CSINHT ne peut pas être utilisé si la valeur de la zone *MDPAT* dans MQMD est ATBRKR.

La valeur initiale de cette zone est CSUNDF.

### **MEENC (entier signé à 10 chiffres)**

MEENC (entier signé à 10 chiffres)

Indique le codage numérique des données qui suivent la structure MQMDE ; il ne s'applique pas aux données numériques de la structure MQMDE elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Le gestionnaire de files d'attente ne vérifie pas que la zone est valide. Pour plus d'informations sur les codages de données, voir la zone *MDENC* décrite dans [«MQMD \(descripteur de message\) sous IBM i»](#), à la page 1162 .

La valeur initiale de cette zone est ENNAT.

### **MEFLG (entier signé à 10 chiffres)**

Indicateurs généraux.

L'indicateur suivant peut être spécifié:

#### **MEFNON**

Aucun indicateur.

La valeur initiale de cette zone est MEFNON.

### **MEFMT (chaîne de caractères de 8 octets)**

Nom de format des données qui suivent MQMDE.

Indique le nom de format des données qui suivent la structure MQMDE.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données. Le gestionnaire de files d'attente ne vérifie pas que cette zone est valide. Pour plus d'informations sur les noms de format, voir la zone *MDFMT* décrite dans «MQMD (descripteur de message) sous IBM i», à la page 1162.

La valeur initiale de cette zone est FMNONE.

#### **MEGID (chaîne de 24 octets)**

Identificateur de groupe.

Voir la zone *MDGID* décrite dans «MQMD (descripteur de message) sous IBM i», à la page 1162. La valeur initiale de cette zone est GINONE.

#### **MELEN (entier signé à 10 chiffres)**

Longueur de la structure MQMDE.

La valeur suivante est définie:

##### **MELEN2**

Longueur de la structure d'extension du descripteur de message version-2.

La valeur initiale de cette zone est MELEN2.

#### **MEMFL (entier signé à 10 chiffres)**

Indicateurs de message.

Voir la zone *MDMFL* décrite dans «MQMD (descripteur de message) sous IBM i», à la page 1162. La valeur initiale de cette zone est MFNONE.

#### **MEOFF (entier signé à 10 chiffres)**

Décalage des données dans le message physique à partir du début du message logique.

Voir la zone *MDOFF* décrite dans «MQMD (descripteur de message) sous IBM i», à la page 1162. La valeur initiale de cette zone est 0.

#### **MEOLN (entier signé à 10 chiffres)**

Longueur du message d'origine.

Voir la zone *MDOLN* décrite dans «MQMD (descripteur de message) sous IBM i», à la page 1162. La valeur initiale de cette zone est OLUNDF.

#### **MESEQ (entier signé à 10 chiffres)**

Numéro de séquence du message logique dans le groupe.

Voir la zone *MDSEQ* décrite dans «MQMD (descripteur de message) sous IBM i», à la page 1162. La valeur initiale de cette zone est 1.

#### **MESID (chaîne de caractères de 4 octets)**

Identificateur de structure.

La valeur doit être:

##### **MESIDV**

Identificateur de la structure d'extension du descripteur de message.

La valeur initiale de ce champ est MESIDV.

#### **MEVER (entier signé à 10 chiffres)**

Numéro de version de la structure.

La valeur doit être:

##### **MEVER2**

Structure d'extension du descripteur de message Version-2.

La constante suivante indique le numéro de version de la version en cours:

**MEVERC**

Version actuelle de la structure d'extension du descripteur de message.

La valeur initiale de cette zone est MEVER2.

**Valeurs initiales**

Tableau 711. Zones dans MQMDE		
Nom de zone	Nom de la constante	Valeur de la constante
MESID	MESIDV	'MDE↵'
MEVER	MEVER2	2
MELEN	MELEN2	72
MEENC	ENNAT	Dépend de l'environnement
MECSI	CSUNDF	0
MEFMT	FMAUCUN	Espaces vides
MEFLG	MEFNON	0
MEGID	GINONE	Valeurs NULL
MESEQ	Aucun	1
MEOFF	Aucun	0
MEMFL	MFNONE	0
MEOLN	ANNULER	-1
<b>Remarques :</b>		
1. Le symbole ↵ représente un caractère blanc unique.		

**Déclaration RPG**

```

D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQMDE Structure
D*
D* Structure identifier
D MESID          1      4    INZ('MDE ')
D* Structure version number
D MEVER          5      8I 0 INZ(2)
D* Length of MQMDE structure
D MELEN          9     12I 0 INZ(72)
D* Numeric encoding of data that followsMQMDE
D MEENC         13     16I 0 INZ(273)
D* Character-set identifier of data thatfollows MQMDE
D MECSI         17     20I 0 INZ(0)
D* Format name of data that followsMQMDE
D MEFMT         21     28    INZ('      ')
D* General flags
D MEFLG         29     32I 0 INZ(0)
D* Group identifier
D MEGID         33     56    INZ(X'00000000000000-
D                      00000000000000000000-
D                      000000000000')
D* Sequence number of logical messagewithin group
D MESEQ         57     60I 0 INZ(1)
D* Offset of data in physical messagefrom start of logical message
D MEOFF         61     64I 0 INZ(0)
D* Message flags

```

```
D MEMFL          65      68I 0 INZ(0)
D* Length of original message
D MEOLN         69      72I 0 INZ(-1)
```

## IBM i MQMHBO (Message handle to buffer options) sous IBM i

Structure définissant les options de traitement des messages dans la mémoire tampon

### Présentation

**Objectif:** La structure MQMHBO permet aux applications de spécifier des options qui contrôlent la façon dont les mémoires tampon sont produites à partir des descripteurs de message. La structure est un paramètre d'entrée dans l'appel MQMHBUF.

**Jeu de caractères et codage:** les données de MQMHBO doivent figurer dans le jeu de caractères de l'application et dans le codage de l'application (ENNAT).

- «Zones», à la page [1214](#)
- «Valeurs initiales», à la page [1215](#)
- «Déclaration RPG», à la page [1215](#)

### Zones

La structure MQMHBO contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

#### MBOPT (entier signé à 10 chiffres)

Descripteur de message de la structure des options de mémoire tampon-Zone MBOPT.

Ces options contrôlent l'action de MQMHBUF.

Vous devez spécifier l'option suivante:

##### Fonction MBPRRF

Lors de la conversion des propriétés d'un descripteur de message en mémoire tampon, convertissez-les au format MQRFH2 .

Si vous le souhaitez, vous pouvez également spécifier l'option suivante. Pour spécifier plusieurs options, ajoutez les valeurs ensemble (n'ajoutez pas la même constante plusieurs fois) ou combinez les valeurs à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations bit).

##### MBDLPR

Les propriétés ajoutées à la mémoire tampon sont supprimées du descripteur de message. Si l'appel échoue, aucune propriété n'est supprimée.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MBPRRF.

#### MBSID (entier signé à 10 chiffres)

Descripteur de message pour la structure des options de mémoire tampon-Zone MBSID.

Il s'agit de l'identificateur de structure. La valeur doit être:

##### MSIDV

Identificateur de la structure des options de traitement des messages vers la mémoire tampon.

Il s'agit toujours d'une zone d'entrée. Valeur initiale de cette zone isMBSIDV.

#### MBVER (entier signé à 10 chiffres)

Il s'agit du numéro de version de la structure. La valeur doit être:

##### MBVER1

Numéro de version de la structure des options de traitement des messages à la mémoire tampon.

La constante suivante indique le numéro de version de la version en cours:

## MMBVERC

Version actuelle de la structure des options du descripteur de message vers la mémoire tampon.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MMBVER1.

## Valeurs initiales

Tableau 712. Zones dans MQMHBO

Nom de zone	Nom de la constante	Valeur de la constante
MVSID	MSIDV	'MHBO'
MBVER	MBVER1	1
MBOPT	Fonction MBPRRF	

### Remarques :

1. La valeur Chaîne NULL ou blancs indique un caractère blanc.

## Déclaration RPG

```
D* MQMHBO Structure
D*
D*
D* Structure identifier
D MBSID          1      4      INZ('MHBO')
D*
D* Structure version number
D MBVER          5      8I 0 INZ(1)
D*
D* Options that control the action of MQMHBUF
D MBOPT          9      12I 0 INZ(1)
```

## IBM i MQOD (descripteur d'objet) sous IBM i

La structure MQOD est utilisée pour spécifier un objet par nom.

### Présentation

**Objectif:** Les types d'objet suivants sont valides:

- File d'attente ou liste de distribution
- Liste de noms
- Définition de processus
- Gestionnaire de files d'attente
- Topic

La structure est un paramètre d'entrée-sortie dans les appels MQOPEN et MQPUT1 .

**Version:** la version actuelle de MQOD est ODVER4. Les zones qui existent uniquement dans les versions les plus récentes de la structure sont identifiées comme telles dans les descriptions qui suivent.

Le fichier COPY fourni contient la version la plus récente de MQOD prise en charge par l'environnement, mais avec la valeur initiale de la zone ODVER définie sur ODVER1. Pour utiliser des zones qui ne sont pas présentes dans la structure version-1 , l'application doit définir la zone ODVER sur le numéro de version de la version requise.

Pour ouvrir une liste de distribution, ODVER doit avoir la valeur ODVER2 ou une valeur supérieure.

**Jeu de caractères et codage:** les données dans MQOD doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente CodedCharSetId et dans le codage du gestionnaire de

files d'attente local indiqué par ENNAT. Toutefois, si l'application s'exécute en tant que client IBM MQ , la structure doit être dans le jeu de caractères et le codage du client.

- «Zones», à la page 1216
- «Valeurs initiales», à la page 1223
- «Déclaration RPG», à la page 1224

## Zones

La structure MQOD contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

### **ODASI (chaîne de bits de 40 octets)**

Autre identificateur de sécurité.

Il s'agit d'un identificateur de sécurité qui est transmis avec *ODAU* au service d'autorisation pour permettre l'exécution des vérifications d'autorisation appropriées. *ODASI* est utilisé uniquement si:

- OOALTU est indiqué dans l'appel MQOPEN, ou
- PMALTU est spécifié dans l'appel MQPUT1 ,

et la zone *ODAU* n'est pas entièrement vide jusqu'au premier caractère null ou jusqu'à la fin de la zone.

La zone *ODASI* possède la structure suivante:

- Le premier octet est un entier binaire contenant la longueur des données significatives qui suivent ; la valeur exclut l'octet de longueur lui-même. Si aucun identificateur de sécurité n'est présent, la longueur est égale à zéro.
- Le deuxième octet indique le type d'identificateur de sécurité présent ; les valeurs suivantes sont possibles:

#### **SITWNT**

Identificateur de sécurité Windows .

#### **SITNON**

Aucun identificateur de sécurité.

- Le troisième octet et les suivants jusqu'à la longueur définie par le premier octet contiennent l'identifiant de sécurité lui-même.
- Les octets restants dans la zone sont définis sur zéro binaire.

La valeur spéciale suivante peut être utilisée:

#### **SINONE**

Aucun identificateur de sécurité n'a été indiqué.

La valeur est zéro binaire pour la longueur de la zone.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par LNSCID. La valeur initiale de cette zone est SINONE. Cette zone est ignorée si *ODVER* est inférieur à ODVER3.

### **ODAU (chaîne de caractères de 12 octets)**

Autre ID utilisateur.

Si OOALTU est indiqué pour l'appel MQOPEN ou PMALTU pour l'appel MQPUT1 , cette zone contient un autre identificateur d'utilisateur qui doit être utilisé pour vérifier l'autorisation de l'ouverture, à la place de l'identificateur d'utilisateur sous lequel l'application est en cours d'exécution. Cependant, certaines vérifications sont toujours effectuées avec l'identificateur d'utilisateur en cours (par exemple, des vérifications de contexte).

Si OOALTU et PMALTU ne sont pas indiqués et que cette zone est entièrement à blanc jusqu'au premier caractère NULL ou jusqu'à la fin de la zone, l'ouverture ne peut aboutir que si aucune autorisation utilisateur n'est requise pour ouvrir cet objet avec les options indiquées.

Si vous n'indiquez ni OOALTU ni PMALTU, cette zone est ignorée.



Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par LNUID. La valeur initiale de cette zone est de 12 caractères blancs.

### **ODDN (chaîne de caractères de 48 octets)**

Nom de la file d'attente dynamique.

Il s'agit du nom d'une file d'attente dynamique qui doit être créée par l'appel MQOPEN. Cela n'est pertinent que lorsque *ODON* spécifie le nom d'une file d'attente modèle ; dans tous les autres cas, *ODDN* est ignoré.

Les caractères valides dans le nom sont identiques à ceux de *ODON*, sauf qu'un astérisque est également valide. Un nom à blanc (ou dans lequel seuls les blancs sont affichés avant le premier caractère nul) n'est pas valide si *ODON* est le nom d'une file d'attente modèle.

Si le dernier caractère non blanc du nom est un astérisque (\*), le gestionnaire de files d'attente le remplace par une chaîne de caractères qui garantit que le nom généré pour la file d'attente est unique au niveau du gestionnaire de files d'attente local. Pour autoriser un nombre suffisant de caractères, l'astérisque est admis uniquement aux positions 1 à 33. Il ne doit pas y avoir d'autres caractères que des blancs ou un caractère nul après l'astérisque.

L'astérisque peut apparaître à la première position de caractère, auquel cas le nom se compose uniquement des caractères générés par le gestionnaire de files d'attente.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par LNQN. La valeur initiale de cette zone est 'AMQ.\*', complétée avec des blancs.

### **ODIDC (entier signé à 10 chiffres)**

Nombre de files d'attente dont l'ouverture a échoué.

Il s'agit du nombre de files d'attente de la liste de distribution dont l'ouverture a échoué. Si cette zone est présente, elle est également définie lors de l'ouverture d'une file d'attente unique qui ne figure pas dans une liste de distribution.

**Remarque :** Si cette zone est présente, elle n'est définie que si le paramètre **CMPCOD** de l'appel MQOPEN ou MQPUT1 est CCOK ou CCWARN ; elle n'est pas définie si le paramètre **CMPCOD** est CCFAIL.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0. Cette zone est ignorée si *ODVER* est inférieur à *ODVER2*.

### **ODKDC (entier signé à 10 chiffres)**

Nombre de files d'attente locales ouvertes avec succès.

Il s'agit du nombre de files d'attente de la liste de distribution qui ont été résolues en files d'attente locales et qui ont été ouvertes avec succès. Le nombre n'inclut pas les files d'attente qui se résolvent en files d'attente éloignées (même si une file d'attente de transmission locale est utilisée initialement pour stocker le message). Si cette zone est présente, elle est également définie lors de l'ouverture d'une file d'attente unique qui ne figure pas dans une liste de distribution.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0. Cette zone est ignorée si *ODVER* est inférieur à *ODVER2*.

### **ODMN (chaîne de caractères de 48 octets)**

Nom du gestionnaire de files d'attente d'objet.

Il s'agit du nom du gestionnaire de files d'attente sur lequel l'objet *ODON* est défini. Les caractères valides dans le nom sont identiques à ceux de *ODON* (voir précédemment). Un nom entièrement vide jusqu'au premier caractère NULL ou jusqu'à la fin de la zone indique le gestionnaire de files d'attente auquel l'application est connectée (le gestionnaire de files d'attente local).

Les points suivants s'appliquent aux types d'objet indiqués:

- Si *ODOT* est défini sur OTTOP, OTNLST, OTPRO ou OTQM, *ODMN* doit être à blanc ou le nom du gestionnaire de files d'attente local.

- Si *ODON* est le nom d'une file d'attente modèle, le gestionnaire de files d'attente crée une file d'attente dynamique avec les attributs de la file d'attente modèle et renvoie dans la zone *ODMN* le nom du gestionnaire de files d'attente sur lequel la file d'attente est créée ; il s'agit du nom du gestionnaire de files d'attente local. Une file d'attente modèle ne peut être indiquée que sur l'appel MQOPEN ; une file d'attente modèle n'est pas valide sur l'appel MQPUT1 .
- Si *ODON* est le nom d'une file d'attente de cluster et que *ODMN* est vide, la destination réelle des messages envoyés à l'aide de l'identificateur de file d'attente renvoyé par l'appel MQOPEN est choisie par le gestionnaire de files d'attente (ou l'exit de charge de travail de cluster, s'il en est installé un) comme suit:
  - Si OOBND0 est spécifié, le gestionnaire de files d'attente sélectionne une instance de la file d'attente de cluster lors du traitement de l'appel MQOPEN et tous les messages insérés à l'aide de cet identificateur de file d'attente sont envoyés à cette instance.
  - Si OOBNDN est spécifié, le gestionnaire de files d'attente peut choisir une instance différente de la file d'attente de destination (résidant sur un gestionnaire de files d'attente différent dans le cluster) pour chaque appel MQPUT successif qui utilise cet identificateur de file d'attente.

Si l'application doit envoyer un message à une instance *spécifique* d'une file d'attente de cluster (c'est-à-dire une instance de file d'attente résidant sur un gestionnaire de files d'attente particulier du cluster), l'application doit spécifier le nom de ce gestionnaire de files d'attente dans la zone *ODMN* . Cela force le gestionnaire de files d'attente local à envoyer le message au gestionnaire de files d'attente de destination spécifié.

- Si l'objet ouvert est une liste de distribution (c'est-à-dire que *ODREC* est supérieur à zéro), *ODMN* doit être vide ou la chaîne null. Si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie RC2153.

Il s'agit d'une zone d'entrée-sortie pour l'appel MQOPEN lorsque *ODON* est le nom d'une file d'attente modèle et d'une zone d'entrée uniquement dans tous les autres cas. La longueur de cette zone est indiquée par LNQM. La valeur initiale de cette zone est de 48 caractères blancs.

### **ODON (chaîne de caractères de 48 octets)**

Nom de l'objet.

Il s'agit du nom local de l'objet tel qu'il est défini sur le gestionnaire de files d'attente identifié par *ODMN*. Le nom peut contenir les caractères suivants:

- Caractères alphabétiques en majuscules (A à Z)
- Caractères alphabétiques minuscules (a à z)
- Chiffres (0 à 9)
- Point (.), barre oblique (/), trait de soulignement (\_), pourcentage (%)

Le nom ne doit pas contenir d'espaces de début ou d'espaces imbriqués, mais peut contenir des espaces de fin. Un caractère NULL peut être utilisé pour indiquer la fin des données significatives dans le nom ; la valeur NULL et les caractères qui la suivent sont traités comme des blancs. Les restrictions suivantes s'appliquent dans les environnements indiqués:

- Sur les systèmes qui utilisent EBCDIC Katakana, les caractères minuscules ne peuvent pas être utilisés.
- Sous IBM i, les noms contenant des caractères minuscules, des barres obliques ou des pourcentages doivent être placés entre guillemets lorsqu'ils sont spécifiés dans les commandes. Ces guillemets ne doivent pas être indiqués pour les noms qui apparaissent en tant que zones dans les structures ou en tant que paramètres dans les appels.

Les points suivants s'appliquent aux types d'objet indiqués:

- Si *ODON* est le nom d'une file d'attente modèle, le gestionnaire de files d'attente crée une file d'attente dynamique avec les attributs de la file d'attente modèle et renvoie dans la zone *ODON* le nom de la file d'attente créée. Une file d'attente modèle ne peut être indiquée que sur l'appel MQOPEN ; une file d'attente modèle n'est pas valide sur l'appel MQPUT1 .

- Si l'objet en cours d'ouverture est une liste de distribution (c'est-à-dire si *ODREC* est présent et supérieur à zéro), *ODON* doit être vide ou la chaîne null. Si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie RC2152.
- Si *ODOT* est OTQM, des règles spéciales s'appliquent ; dans ce cas, le nom doit être entièrement vide jusqu'au premier caractère null ou jusqu'à la fin de la zone.
- Si *ODON* est le nom d'une file d'attente alias avec TARGTYPE (TOPIC), un contrôle de sécurité est d'abord effectué sur la file d'attente alias nommée, comme cela est normal pour l'utilisation des files d'attente alias. Si cette vérification de sécurité aboutit, cet appel MQOPEN se poursuit et se comporte comme un MQOPEN d'un OTTOP, y compris une vérification de la sécurité de l'objet de rubrique d'administration.

Il s'agit d'une zone d'entrée-sortie pour l'appel MQOPEN lorsque *ODON* est le nom d'une file d'attente modèle et d'une zone d'entrée uniquement dans tous les autres cas. La longueur de cette zone est indiquée par LNQN. La valeur initiale de cette zone est de 48 caractères blancs.

Le nom complet de la rubrique peut être généré à partir de deux zones différentes: *ODON* et *ODOS*. Pour plus de détails sur l'utilisation de ces deux zones, voir [Combinaison de chaînes de rubrique](#).

### **ODORO (entier signé à 10 chiffres)**

Décalage du premier enregistrement d'objet à partir du début de MQOD.

Décalage en octets du premier enregistrement d'objet MQOR à partir du début de la structure MQOD. Le décalage peut être positif ou négatif. *ODORO* est utilisé uniquement lorsqu'une liste de distribution est ouverte. La zone est ignorée si *ODREC* a la valeur zéro.

Lorsqu'une liste de distribution est ouverte, un tableau d'un ou de plusieurs enregistrements d'objet MQOR doit être fourni afin de spécifier les noms des files d'attente de destination dans la liste de distribution. Cette opération peut être effectuée de l'une des deux manières suivantes:

- En utilisant la zone de décalage *ODORO*

Dans ce cas, l'application doit déclarer sa propre structure contenant un MQOD suivi du tableau d'enregistrements MQOR (avec autant d'éléments de tableau que nécessaire) et définir *ODORO* sur le décalage du premier élément du tableau à partir du début du MQOD. Veillez à ce que ce décalage soit correct.

- A l'aide de la zone de pointeur *ODORP*

Dans ce cas, l'application peut déclarer le tableau de structures MQOR séparément de la structure MQOD et définir *ODORP* sur l'adresse du tableau.

Quelle que soit la technique choisie, *ODORO* et *ODORP* doivent être utilisés ; l'appel échoue avec le code anomalie RC2155 si les deux sont égaux à zéro ou si les deux sont différents de zéro.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *ODVER* est inférieur à ODVER2.

### **ODORP (pointeur)**

Adresse du premier enregistrement d'objet.

Il s'agit de l'adresse du premier enregistrement d'objet MQOR. *ODORP* est utilisé uniquement lorsqu'une liste de distribution est ouverte. La zone est ignorée si *ODREC* a la valeur zéro.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null. *ODORP* ou *ODORO* peut être utilisé pour spécifier les enregistrements d'objet, mais pas les deux ; voir la description de la zone *ODORO* précédemment pour plus de détails. Si *ODORP* n'est pas utilisé, il doit être défini sur le pointeur null ou sur les octets null. Cette zone est ignorée si *ODVER* est inférieur à ODVER2.

### **ODOS (MQCHARV)**

ODOS indique le nom d'objet long à utiliser.

Cette zone est référencée uniquement pour certaines valeurs de *ODOT*. Voir la description de [ODOT](#) pour plus de détails sur les valeurs qui indiquent que cette zone est utilisée.

Si *ODOS* est spécifié de manière incorrecte, conformément à la description de l'utilisation de la structure *MQCHARV*, ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie RC2441.

Il s'agit d'une zone d'entrée. Les valeurs initiales des zones de cette structure sont identiques à celles de la structure *MQCHARV*.

Le nom complet de la rubrique peut être généré à partir de deux zones différentes: *ODON* et *ODOS*. Pour plus de détails sur l'utilisation de ces deux zones, voir [Combinaison de chaînes de rubrique](#). Cette zone est ignorée si *ODVER* est inférieur à *ODVER4*.

### **ODOT (entier signé à 10 chiffres)**

Type d'objet.

Type d'objet nommé dans *ODON*. Les valeurs possibles sont les suivantes :

#### **File d'attente OTQ**

File d'attente. Le nom de l'objet se trouve dans *ODON*.

#### **OTNLST**

NAMELIST. Le nom de l'objet se trouve dans *ODON*.

#### **OTPRO**

Définition de processus. Le nom de l'objet se trouve dans *ODON*.

#### **OTQM**

Gestionnaire de files d'attente. Le nom de l'objet se trouve dans *ODON*.

#### **Mot de passe OTTOP**

:NONE. Le nom complet de la rubrique peut être généré à partir de deux zones différentes: *ODON* et *ODOS*.

Pour plus de détails sur l'utilisation de ces deux zones, voir [Combinaison de chaînes de rubrique](#).

Si l'objet identifié par la zone *ODON* est introuvable, l'appel échoue avec le code anomalie RC2425 même si une chaîne est spécifiée dans *ODOS*.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est *OTQ*.

### **ODREC (entier signé à 10 chiffres)**

Nombre d'enregistrements d'objet présents.

Il s'agit du nombre d'enregistrements d'objet *MQOR* qui ont été fournis par l'application. Si ce nombre est supérieur à zéro, cela indique qu'une liste de distribution est en cours d'ouverture, *ODREC* étant le nombre de files d'attente de destination dans la liste. Une liste de distribution ne peut contenir qu'une seule destination.

La valeur de *ODREC* ne doit pas être inférieure à zéro, et si elle est supérieure à zéro, *ODOT* doit être *OTQ* ; l'appel échoue avec le code anomalie RC2154 si ces conditions ne sont pas remplies.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *ODVER* est inférieur à *ODVER2*.

### **ODRMN (chaîne de caractères de 48 octets)**

Nom du gestionnaire de files d'attente résolu.

Il s'agit du nom du gestionnaire de files d'attente de destination une fois la résolution de nom effectuée par le gestionnaire de files d'attente local. Le nom renvoyé est le nom du gestionnaire de files d'attente qui possède la file d'attente identifiée par *ODRQN*. *ODRMN* peut être le nom du gestionnaire de files d'attente local.

Si *ODRQN* est une file d'attente partagée appartenant au groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local, *ODRMN* est le nom du groupe de partage de files d'attente. Si la file d'attente appartient à un autre groupe de partage de files d'attente, *ODRQN* peut être le nom du groupe de partage de files d'attente ou le nom d'un gestionnaire de files d'attente membre du groupe de partage de files d'attente (la nature de la valeur renvoyée est déterminée par les définitions de files d'attente qui existent sur le gestionnaire de files d'attente local).

Une valeur non vide est renvoyée uniquement si l'objet est une file d'attente unique ouverte pour l'exploration, l'entrée ou la sortie (ou toute combinaison). Si l'objet ouvert est l'un des suivants, *ODRMN* est mis à blanc:

- N'est pas une file d'attente
- Une file d'attente, mais non ouverte pour l'exploration, l'entrée ou la sortie
- Une file d'attente de cluster avec OOBNDN spécifié (ou avec OOBNDQ en vigueur lorsque l'attribut de file d'attente **DefBind** a la valeur BNDNOT)
- Une liste de distribution

Il s'agit d'une zone de sortie. La longueur de cette zone est indiquée par LNQN. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation. Cette zone est ignorée si *ODVER* est inférieur à ODVER3.

### **ODRO (MQCHARV)**

ODRO est le nom d'objet long après que le gestionnaire de files d'attente a résolu le nom fourni dans *ODON*.

Cette zone est renvoyée uniquement pour certains types d'objets, de rubriques et d'alias de file d'attente qui font référence à un objet de rubrique.

Si le nom d'objet long est fourni dans *ODOS* et que rien n'est fourni dans *ODON*, la valeur renvoyée dans cette zone est la même que celle fournie dans *ODOS*.

Si cette zone est omise (c'est-à-dire ODRO.VSBufSize est égal à zéro), *ODRO* n'est pas renvoyé, mais la longueur est renvoyée dans ODRO.VSLength. Si la longueur est inférieure à la longueur *ODRO* complète, elle est tronquée et renvoie autant de caractères les plus à droite que la longueur indiquée.

Si *ODRO* est spécifié de manière incorrecte, conformément à la description de l'utilisation de la structure MQCHARV, ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie RC2520. Cette zone est ignorée si *ODVER* est inférieur à ODVER4.

### **ODRQN (chaîne de caractères de 48 octets)**

Nom de file d'attente résolu.

Il s'agit du nom de la file d'attente de destination une fois la résolution de nom effectuée par le gestionnaire de files d'attente local. Le nom renvoyé est le nom d'une file d'attente qui existe sur le gestionnaire de files d'attente identifié par *ODRMN*.

Une valeur non vide est renvoyée uniquement si l'objet est une file d'attente unique ouverte pour l'exploration, l'entrée ou la sortie (ou toute combinaison). Si l'objet ouvert est l'un des suivants, *ODRQN* est mis à blanc:

- N'est pas une file d'attente
- Une file d'attente, mais non ouverte pour l'exploration, l'entrée ou la sortie
- Une liste de distribution
- Une file d'attente alias qui fait référence à un objet de rubrique (voir «ODRO (MQCHARV)», à la page 1221 à la place)

Il s'agit d'une zone de sortie. La longueur de cette zone est indiquée par LNQN. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation. Cette zone est ignorée si *ODVER* est inférieur à ODVER3.

### **ODRRO (entier signé à 10 chiffres)**

Décalage du premier enregistrement de réponse à partir du début de MQOD.

Décalage en octets du premier enregistrement de réponse MQRR à partir du début de la structure MQOD. Le décalage peut être positif ou négatif. *ODRRO* est utilisé uniquement lorsqu'une liste de distribution est ouverte. La zone est ignorée si *ODREC* a la valeur zéro.

Lorsqu'une liste de distribution est ouverte, un tableau d'un ou de plusieurs enregistrements de réponse MQRR peut être fourni afin d'identifier les files d'attente dont l'ouverture a échoué (zoneRRCC

dans MQRR) et la raison de chaque échec (zoneRRREA dans MQRR). Les données sont renvoyées dans le tableau des enregistrements de réponse dans le même ordre que les noms de file d'attente dans le tableau des enregistrements d'objet. Le gestionnaire de files d'attente définit les enregistrements de réponse uniquement lorsque le résultat de l'appel est mixte (c'est-à-dire que certaines files d'attente ont été ouvertes avec succès alors que d'autres ont échoué ou que toutes ont échoué, mais pour des raisons différentes) ; le code anomalie RC2136 de l'appel indique ce cas. Si le même code anomalie s'applique à toutes les files d'attente, ce code anomalie est renvoyé dans le paramètre **REASON** de l'appel MQOPEN ou MQPUT1 et les enregistrements de réponse ne sont pas définis. Les enregistrements de réponse sont facultatifs, mais s'ils sont fournis, ils doivent être *ODREC* .

Les enregistrements de réponse peuvent être fournis de la même manière que les enregistrements d'objet, soit en spécifiant un décalage dans *ODRRO*, soit en spécifiant une adresse dans *ODRRP* ; voir la description de *ODORO* précédemment pour plus de détails sur la procédure à suivre. Toutefois, vous ne pouvez pas utiliser plus d'un des codes *ODRRO* et *ODRRP* ; l'appel échoue avec le code anomalie RC2156 si les deux sont différents de zéro.

Pour l'appel MQPUT1 , ces enregistrements de réponse sont utilisés pour renvoyer des informations sur les erreurs qui se produisent lorsque le message est envoyé aux files d'attente de la liste de distribution, ainsi que sur les erreurs qui se produisent lorsque les files d'attente sont ouvertes. Le code achèvement et le code raison de l'opération d'insertion pour une file d'attente remplacent ceux de l'opération d'ouverture pour cette file d'attente uniquement si le code achèvement de cette dernière était CCOK ou CCWARN.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *ODVER* est inférieur à ODVER2.

#### **ODRRP (pointeur)**

Adresse du premier enregistrement de réponse.

Il s'agit de l'adresse du premier enregistrement de réponse MQRR. *ODRRP* est utilisé uniquement lorsqu'une liste de distribution est ouverte. La zone est ignorée si *ODREC* a la valeur zéro.

*ODRRP* ou *ODRRO* peut être utilisé pour spécifier les enregistrements de réponse, mais pas les deux ; voir la description précédente de la zone *ODRRO* pour plus de détails. Si *ODRRP* n'est pas utilisé, il doit être défini sur le pointeur null ou sur les octets null.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null. Cette zone est ignorée si *ODVER* est inférieur à ODVER2.

#### **ODSID (chaîne de caractères de 4 octets)**

Identificateur de structure.

La valeur doit être:

##### **ODSIDV**

Identificateur de la structure de descripteur d'objet.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est ODSIDV.

#### **ODSS (MQCHARV)**

ODSS contient la chaîne utilisée pour fournir les critères de sélection utilisés lors de l'extraction de messages d'une file d'attente.

*ODSS* ne doit pas être fourni dans les cas suivants:

- Si *ODOT* n'est pas OTQ
- Si la file d'attente en cours d'ouverture n'est pas ouverte à l'aide de l'une des options d'entrée, OOINP\*

Si *ODSS* est fourni dans ces cas, l'appel échoue avec le code anomalie RC2516.

Si *ODSS* est spécifié de manière incorrecte, conformément à la description de l'utilisation de la structure MQCHARV , ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie RC2519. Cette zone est ignorée si *ODVER* est inférieur à ODVER4.

### **ODUDC (entier signé à 10 chiffres)**

Nombre de files d'attente éloignées ouvertes correctement

Il s'agit du nombre de files d'attente de la liste de distribution qui sont résolues en files d'attente éloignées et qui ont été ouvertes avec succès. Si cette zone est présente, elle est également définie lors de l'ouverture d'une file d'attente unique qui ne figure pas dans une liste de distribution.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0. Cette zone est ignorée si *ODVER* est inférieur à *ODVER2*.

### **ODVER (entier signé à 10 chiffres)**

Numéro de version de la structure.

Il doit s'agir de l'une des valeurs suivantes :

#### **ODVER1**

Structure de descripteur d'objet Version-1 .

#### **ODVER2**

Structure de descripteur d'objet Version-2 .

#### **ODVER3**

Structure de descripteur d'objet Version-3 .

#### **ODVER4**

Structure de descripteur d'objet Version-4 .

Les zones qui existent uniquement dans les versions plus récentes de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

#### **ODVERC**

Version actuelle de la structure de descripteur d'objet.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est *ODVER1*.

### **Valeurs initiales**

<b>Nom de zone</b>	<b>Nom de la constante</b>	<b>Valeur de la constante</b>
<i>ODSID</i>	ODSIDV	'ODSID'
<i>ODVER</i>	ODVER1	1
<i>ODOT</i>	File d'attente OTQ	1
<i>ODON</i>	Aucun	Espaces vides
<i>ODMN</i>	Aucun	Espaces vides
<i>ODDN</i>	Aucun	'AMQ.*'
<i>ODAU</i>	Aucun	Espaces vides
<i>ODREC</i>	Aucun	0
<i>ODKDC</i>	Aucun	0
<i>ODUDC</i>	Aucun	0
<i>ODIDC</i>	Aucun	0
<i>ODORO</i>	Aucun	0
<i>ODRRO</i>	Aucun	0

Tableau 713. Zones dans MQOD (suite)

Nom de zone	Nom de la constante	Valeur de la constante
ODORP	Aucun	Pointeur null ou octets null
ODRRP	Aucun	Pointeur null ou octets null
ODASI	SINONE	Valeurs NULL
ODRQN	Aucun	Espaces vides
ODRMN	Aucun	Espaces vides
ODOS	Comme défini pour MQCHARV	Comme défini pour MQCHARV
ODRO	Comme indiqué dans ODOS	Comme indiqué dans ODOS
ODSS	Aucun	Espaces vides

**Remarques :**

1. Le symbole ~ représente un caractère blanc unique.

## Déclaration RPG

```

D*.1.....2.....3.....4.....5.....6.....7..
D*
D* MQOD Structure
D*
D*
D* Structure identifier
D ODSID          1      4    INZ('OD ')
D*
D* Structure version number
D ODVER          5      8I 0 INZ(1)
D*
D* Object type
D ODOT           9     12I 0 INZ(1)
D*
D* Object name
D ODON          13     60    INZ
D*
D* Object queue manager name
D ODMN          61     108   INZ
D*
D* Dynamic queue name
D ODDN          109    156   INZ('AMQ.*')
D*
D* Alternate user identifier
D ODAU          157    168   INZ
D*
** Number of object records
D* present
D ODREC         169    172I 0 INZ(0)
D*
** Number of local queues opened
D* successfully
D ODKDC         173    176I 0 INZ(0)
D*
** Number of remote queues opened
D* successfully
D ODUDC         177    180I 0 INZ(0)
D*
** Number of queues that failed to
D* open
D ODIDC         181    184I 0 INZ(0)
D*
** Offset of first object record
D* from start of MQOD
D ODORO         185    188I 0 INZ(0)
D*
** Offset of first response record
D* from start of MQOD

```



```

D ODRRO                189    192I 0 INZ(0)
D*
D* Address of first object record
D ODORP                193    208*  INZ(*NULL)
D*
D* Address of first response
D* record
D ODORP                209    224*  INZ(*NULL)
D*
D* Alternate security identifier
D ODASI                225    264    INZ(X'0000000000000000-
D                          000000000000000000000000-
D                          000000000000000000000000-
D                          000000000000')
D*
D* Resolved queue name
D ODRQN                265    312    INZ
D*
D* Resolved queue manager name
D ODRMN                313    360    INZ
D*
D* reserved field
D ODRE1                361    364I 0 INZ(0)
D*
D* reserved field
D ODRS2                365    368I 0 INZ(0)
D*
D* Object long name
D* Address of variable length string
D ODOSCHRP             369    384*  INZ(*NULL)
D* Offset of variable length string
D ODOSCHRO             385    388I 0 INZ(0)
D* Size of buffer
D ODOSVSBS             389    392I 0 INZ(-1)
D* Length of variable length string
D ODOSCHRL             393    396I 0 INZ(0)
D* CCSID of variable length string
D ODOSCHRC             397    400I 0 INZ(-3)
D*
D* Message Selector
D* Address of variable length string
D ODSSCHRP             401    416*  INZ(*NULL)
D* Offset of variable length string
D ODSSCHRO             417    420I 0 INZ(0)
D* Size of buffer
D ODSSVSBS             421    424I 0 INZ(-1)
D* Length of variable length string
D ODSSCHRL             425    428I 0 INZ(0)
D* CCSID of variable length string
D ODSSCHRC             429    432I 0 INZ(-3)
D*
D* Resolved long object name
D* Address of variable length string
D ODRSOCHRP            433    448*  INZ(*NULL)
D* Offset of variable length string
D ODRSOCHRO            449    452I 0 INZ(0)
D* Size of buffer
D ODRSOVSBS            453    456I 0 INZ(-1)
D* Length of variable length string
D ODRSOCHRL            457    460I 0 INZ(0)
D* CCSID of variable length string
D ODRSOCHRC            461    464I 0 INZ(-3)
D*
D* Alias queue resolved object type
D ODRT                 465    468I 0 INZ(0)

```

## MQOR (enregistrement d'objet) sur IBM i

La structure MQOR permet de spécifier le nom de file d'attente et le nom de gestionnaire de files d'attente d'une file d'attente de destination unique.

### Présentation

**Objectif:** MQOR est une structure d'entrée pour les appels MQOPEN et MQPUT1 .

**Jeu de caractères et codage:** les données dans MQOR doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et le codage du gestionnaire de files

d'attente local indiqué par ENNAT. Toutefois, si l'application s'exécute en tant que client IBM MQ , la structure doit être dans le jeu de caractères et le codage du client.

**Syntaxe:** en fournissant un tableau de ces structures sur l'appel MQOPEN, il est possible d'ouvrir une liste de files d'attente ; cette liste est appelée *liste de distribution*. Chaque insertion de message à l'aide de l'identificateur de file d'attente renvoyé par cet appel MQOPEN est placée dans chacune des files d'attente de la liste, si l'ouverture de la file d'attente a abouti.

- «Zones», à la page 1226
- «Valeurs initiales», à la page 1226
- «Déclaration RPG», à la page 1226

## Zones

La structure MQOR contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

### ORMN (chaîne de caractères de 48 octets)

Nom du gestionnaire de files d'attente d'objet.

Il s'agit de la même zone que la zone *ODMN* dans la structure MQOD (voir MQOD pour plus de détails).

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est de 48 caractères blancs.

### ORON (chaîne de caractères de 48 octets)

Nom de l'objet.

Il s'agit de la même zone que la zone *ODON* dans la structure MQOD (voir MQOD pour plus de détails), sauf que:

- Il doit s'agir du nom d'une file d'attente.
- Il ne doit pas s'agir du nom d'une file d'attente modèle.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est de 48 caractères blancs.

## Valeurs initiales

Tableau 714. Zones dans MQOR		
Nom de zone	Nom de la constante	Valeur de la constante
ORON	Aucun	Espaces vides
ORMN	Aucun	Espaces vides

## Déclaration RPG

```

D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQOR Structure
D*
D* Object name
D  ORON                1      48    INZ
D* Object queue manager name
D  ORMN                49     96    INZ

```

## MQPD-Descripteur de propriété

**MQPD** est utilisé pour définir les attributs d'une propriété.

## Présentation

**Objectif:** La structure est un paramètre d'entrée-sortie sur l'appel MQSETMP et un paramètre de sortie sur l'appel MQINQMP.

**Jeu de caractères et codage:** les données dans MQPD doivent être dans le jeu de caractères de l'application et le codage de l'application (ENNAT).

- «Zones», à la page [1227](#)
- «Valeurs initiales», à la page [1229](#)
- «Déclaration RPG», à la page [1229](#)

## Zones

La structure MQPD contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

### PDCT (entier signé à 10 chiffres)

Décrit le contexte de message auquel appartient la propriété.

Lorsqu'un gestionnaire de files d'attente reçoit un message contenant une propriété définie par IBM MQ que le gestionnaire de files d'attente reconnaît comme étant incorrecte, le gestionnaire de files d'attente corrige la valeur de la zone *PDCT*.

L'option suivante peut être spécifiée:

#### PDUSC

La propriété est associée au contexte utilisateur.

Aucune autorisation spéciale n'est requise pour pouvoir définir une propriété associée au contexte utilisateur à l'aide de l'appel MQSETMP.

Si l'option décrite précédemment n'est pas requise, l'option suivante peut être utilisée:

#### PDNOC

La propriété n'est pas associée à un contexte de message.

Une valeur non reconnue est rejetée avec le code *PDREA* RC2482.

Il s'agit d'une zone d'entrée-sortie de l'appel MQSETMP et d'une zone de sortie de l'appel MQINQMP. La valeur initiale de cette zone est PDNOC.

### PDCPYOPT (entier signé à 10 chiffres)

Décrit le type de message dans lequel la propriété doit être copiée.

Il s'agit d'un champ de sortie uniquement pour les propriétés définies par IBM MQ reconnues ; IBM MQ définit la valeur appropriée.

Lorsqu'un gestionnaire de files d'attente reçoit un message contenant une propriété définie par IBM MQ que le gestionnaire de files d'attente reconnaît comme étant incorrecte, le gestionnaire de files d'attente corrige la valeur de la zone *CopyOptions*.

Vous pouvez spécifier une ou plusieurs de ces options. Pour spécifier plusieurs options, ajoutez les valeurs ensemble (n'ajoutez pas la même constante plusieurs fois) ou combinez les valeurs à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations bit).

#### COPFOR

Cette propriété est copiée dans un message en cours de transfert.

#### COPPUB

Cette propriété est copiée dans le message reçu par un abonné lors de la publication d'un message.

#### COPREP

Cette propriété est copiée dans un message de réponse.

**PRPC**

Cette propriété est copiée dans un message de rapport.

**COPALL**

Cette propriété est copiée dans tous les types de messages suivants.

**COPNON**

Cette propriété n'est pas copiée dans un message.

**Option par défaut:** L'option suivante peut être spécifiée pour fournir l'ensemble d'options de copie par défaut:

**COPDEF**

Cette propriété est copiée dans un message en cours de transfert, dans un message de rapport ou dans un message reçu par un abonné lors de la publication d'un message.

Cela revient à spécifier la combinaison des options COPFOR, plus COPRP, plus COPPUB.

Si aucune des options décrites précédemment n'est requise, utilisez l'option suivante:

**COPNON**

Utilisez cette valeur pour indiquer qu'aucune autre option de copie n'a été spécifiée ; à l'aide d'un programme, aucune relation n'existe entre cette propriété et les messages suivants. Cette valeur est toujours renvoyée pour les propriétés de descripteur de message.

Il s'agit d'une zone d'entrée-sortie de l'appel MQSETMP et d'une zone de sortie de l'appel MQINQMP. La valeur initiale de cette zone est COPDEF.

**PDOPT (entier signé à 10 chiffres)**

La valeur doit être:

**PDAUCUN**

Aucune option spécifiée

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est PDNONE.

**PDSID (entier signé à 10 chiffres)**

Il s'agit de l'identificateur de structure ; la valeur doit être:

**PSIDV**

Identificateur de la structure de descripteur de propriété.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **PSIDV**.

**PDSUP (entier signé à 10 chiffres)**

Cette zone indique le niveau de prise en charge de la propriété de message requis par le gestionnaire de files d'attente pour que le message contenant cette propriété soit inséré dans une file d'attente. S'applique uniquement aux propriétés définies par IBM MQ; la prise en charge de toutes les autres propriétés est facultative.

La zone est automatiquement définie sur la valeur correcte lorsque la propriété définie par IBM MQ est connue du gestionnaire de files d'attente. Si la propriété n'est pas reconnue, PDSUPO est affecté. Lorsqu'un gestionnaire de files d'attente reçoit un message contenant une propriété définie par IBM MQ que le gestionnaire de files d'attente reconnaît comme étant incorrecte, le gestionnaire de files d'attente corrige la valeur de la zone *PDSUP*.

Lors de la définition d'une propriété définie par IBM MQ à l'aide de l'appel MQSETMP sur un descripteur de message dans lequel l'option CMNOVA a été définie, *PDSUP* devient une zone d'entrée. Cela permet à une application d'insérer une propriété définie par IBM MQ, avec la valeur correcte, dans laquelle la propriété n'est pas prise en charge par le gestionnaire de files d'attente connecté, mais dans laquelle le message doit être traité sur un autre gestionnaire de files d'attente.

La valeur PDSUPO est toujours affectée aux propriétés qui ne sont pas des propriétés définies par IBM MQ.

L'une des valeurs suivantes est renvoyée par l'appel MQINQMP ou l'une des valeurs peut être spécifiée lors de l'utilisation de l'appel MQSETMP sur un descripteur de message dans lequel l'option CMNOVA est définie:

#### **PDSUPO**

La propriété est acceptée par un gestionnaire de files d'attente même si elle n'est pas prise en charge. La propriété peut être supprimée pour que le message soit transmis à un gestionnaire de files d'attente qui ne prend pas en charge les propriétés de message. Cette valeur est également affectée aux propriétés qui ne sont pas définies par IBM MQ.

#### **PDSUPR**

La prise en charge de la propriété est requise. Le message est rejeté par un gestionnaire de files d'attente qui ne prend pas en charge la propriété définie par IBM MQ. L'appel MQPUT ou MQPUT1 échoue avec le code achèvement CCFAIL et le code anomalie RC2490.

#### **PDSUPL**

Le message est rejeté par un gestionnaire de files d'attente qui ne prend pas en charge la propriété définie par IBM MQ si le message est destiné à une file d'attente locale. L'appel MQPUT ou MQPUT1 échoue avec le code achèvement CCFAIL et le code anomalie RC2490.

L'appel MQPUT ou MQPUT1 aboutit si le message est destiné à un gestionnaire de files d'attente éloignées.

Il s'agit d'une zone de sortie sur l'appel MQINQMP et d'une zone d'entrée sur l'appel MQSETMP si le descripteur de message a été créé avec l'option CMNOVA définie. La valeur initiale de cette zone est PDSUPO.

#### **PDVER (entier signé à 10 chiffres)**

Il s'agit du numéro de version de la structure ; la valeur doit être:

##### **PDVER1**

Structure de descripteur de propriété Version-1 .

La constante suivante indique le numéro de version de la version en cours:

##### **PDVERC**

Version actuelle de la structure de descripteur de propriété.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **PDVER1**.

### **Valeurs initiales**

<i>Tableau 715. Zones dans MQPD</i>		
<b>Nom de zone</b>	<b>Nom de la constante</b>	<b>Valeur de la constante</b>
<i>PDSID</i>	PSIDV	'PD'
<i>PDVER</i>	PDVER1	1
<i>PDOPT</i>	PDAUCUN	0
<i>PDSUP</i>	PDSUPO	0
<i>PDCT</i>	PDNOC	0
<i>PDOPYOPT</i>	COPDEF	0

### **Déclaration RPG**

D\* MQDMHO Structure  
D\*

```

D*
D* Structure identifier
D  DMSID          1      4    INZ('DMHO')
D*
D* Structure version number
D  DMVER          5      8I 0 INZ(1)
D*
D* Options that control the action of MQDLTMH
D  DMOPT          9      12I 0 INZ(0)

```

## IBM i MQPMO (options d'insertion de message) sous IBM i

La structure MQPMO permet à l'application de spécifier des options qui contrôlent la façon dont les messages sont placés dans des files d'attente ou publiés dans des rubriques.

### Présentation

#### Objet

La structure est un paramètre d'entrée-sortie dans les appels MQPUT et MQPUT1 .

#### Version

La version actuelle de MQPMO est PMVER2. Les zones qui existent uniquement dans les versions les plus récentes de la structure sont identifiées comme telles dans les descriptions qui suivent.

Le fichier COPY fourni contient la version la plus récente de MQPMO qui est prise en charge par l'environnement, mais avec la valeur initiale de la zone *PMVER* définie sur PMVER1. Pour utiliser des zones qui ne sont pas présentes dans la structure version-1 , l'application doit définir la zone *PMVER* sur le numéro de version de la version requise.

#### Jeu de caractères et codage

Les données de MQPMO doivent se trouver dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par ENNAT. Toutefois, si l'application s'exécute en tant que client IBM MQ , la structure doit être dans le jeu de caractères et le codage du client.

- «Zones», à la page [1230](#)
- «Valeurs initiales», à la page [1245](#)
- «Déclaration RPG», à la page [1245](#)

### Zones

La structure MQPMO contient les zones suivantes. Les zones sont décrites par ordre alphabétique:

#### PMCT (entier signé à 10 chiffres)

Descripteur d'objet de la file d'attente d'entrée.

Si PMPASI ou PMPASA est spécifié, cette zone doit contenir le descripteur de file d'attente d'entrée à partir duquel les informations de contexte à associer au message en cours d'insertion sont extraites.

Si PMPASI et PMPASA ne sont pas indiqués, cette zone est ignorée.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

#### PMIDC (entier signé à 10 chiffres)

Nombre de messages qui n'ont pas pu être envoyés.

Il s'agit du nombre de messages qui n'ont pas pu être envoyés aux files d'attente de la liste de distribution. Le nombre inclut les files d'attente dont l'ouverture a échoué et les files d'attente dont l'ouverture a abouti mais pour lesquelles l'opération d'insertion a échoué. Cette zone est également définie lors de l'insertion d'un message dans une file d'attente unique qui ne figure pas dans une liste de distribution.

**Remarque :** Cette zone est définie uniquement si le paramètre **CMPCOD** de l'appel MQPUT ou MQPUT1 est CCOK ou CCWARN ; elle n'est pas définie si le paramètre **CMPCOD** est CCFAIL.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0. Cette zone n'est pas définie si *PMVER* est inférieur à *PMVER2*.

### **PMKDC (entier signé à 10 chiffres)**

Nombre de messages envoyés avec succès aux files d'attente locales.

Il s'agit du nombre de messages que l'appel MQPUT ou MQPUT1 a envoyés avec succès aux files d'attente de la liste de distribution qui sont des files d'attente locales. Ce nombre n'inclut pas les messages envoyés à des files d'attente qui se résolvent en files d'attente éloignées (même si une file d'attente de transmission locale est utilisée initialement pour stocker le message). Cette zone est également définie lors de l'insertion d'un message dans une file d'attente unique qui ne figure pas dans une liste de distribution.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0. Cette zone n'est pas définie si *PMVER* est inférieur à *PMVER2*.

### **PMOPT (entier signé à 10 chiffres)**

Options qui contrôlent l'action de MQPUT et de MQPUT1.

Vous pouvez indiquer tout ou partie des éléments suivants. Si plusieurs valeurs sont requises, les valeurs peuvent être ajoutées (n'ajoutez pas la même constante plus d'une fois). Les combinaisons qui ne sont pas valides sont notées ; les autres combinaisons sont valides.

**Options de publication:** Les options suivantes contrôlent la manière dont les messages sont publiés dans une rubrique.

#### **PPMSRTO**

Toute information renseignée dans les champs MDRQ et MDRM du MQMD de cette publication n'est pas transmise aux abonnés. Si cette option est utilisée avec une option de rapport qui requiert une file d'attente ReplyTo, l'appel échoue avec RC2027 .

#### **PMRET**

La publication envoyée doit être conservée par le gestionnaire de files d'attente. Cela permet à un abonné de demander une copie de cette publication après sa publication, à l'aide de l'appel MQSUBRQ. Il permet également d'envoyer une publication aux applications qui s'abonnent après la publication, à moins qu'elles ne choisissent de ne pas l'envoyer à l'aide de l'option SONEWP. Si une application reçoit une publication qui a été conservée, elle est indiquée par la propriété de message mq.IsRetained de cette publication.

Une seule publication peut être conservée sur chaque noeud de l'arborescence de rubriques. Cela signifie que s'il existe déjà une publication conservée pour cette rubrique, publiée par toute autre application, elle est remplacée par cette publication. Il est donc préférable d'éviter que plus d'un diffuseur de publications conserve des messages sur le même sujet.

Lorsque des publications conservées sont demandées par un abonné, l'abonnement utilisé peut contenir un caractère générique dans la rubrique, auquel cas un certain nombre de publications conservées peuvent correspondre (sur différents noeuds de l'arborescence de rubriques) et plusieurs publications peuvent être envoyées à l'application demandeuse. Pour plus de détails, voir la description de l'appel «MQSUBRQ-Demande d'abonnement», à la page 832 .

Si cette option est utilisée et que la publication ne peut pas être conservée, le message n'est pas publié et l'appel échoue avec RC2479 .

**Options de point de synchronisation:** Les options suivantes concernent la participation de l'appel MQPUT ou MQPUT1 dans une unité de travail:

#### **PPMSYP**

Insérer un message avec le contrôle de point de synchronisation.

La demande consiste à fonctionner dans le cadre des protocoles d'unité de travail normaux. Le message n'est pas visible en dehors de l'unité de travail tant que l'unité de travail n'est pas validée. Si l'unité d'oeuvre est annulée, le message est supprimé.

Si cette option et PMNSYP ne sont pas spécifiées, la demande d'insertion ne se trouve pas dans une unité d'oeuvre.

PMSYP ne doit pas être indiqué avec PMNSYP.

### **PMNSYP**

Message d'insertion sans contrôle de point de synchronisation.

La demande consiste à fonctionner en dehors des protocoles d'unité de travail normaux. Le message est disponible immédiatement et ne peut pas être supprimé par l'annulation d'une unité de travail.

Si cette option et PMSYP ne sont pas spécifiées, la demande d'insertion ne fait pas partie d'une unité d'oeuvre.

PMNSYP ne doit pas être indiqué avec PMSYP.

**Options d'identificateur de message et d'identificateur de corrélation:** Les options suivantes demandent au gestionnaire de files d'attente de générer un nouvel identificateur de message ou un nouvel identificateur de corrélation:

### **ID PMN**

Générez un nouvel identificateur de message.

Cette option permet au gestionnaire de files d'attente de remplacer le contenu de la zone *MDMID* dans MQMD par un nouvel identificateur de message. Cet identificateur de message est envoyé avec le message et renvoyé à l'application dans la sortie de l'appel MQPUT ou MQPUT1 .

Cette option peut également être spécifiée lorsque le message est inséré dans une liste de distribution. Pour plus de détails, voir la description de la zone *PRMID* dans la structure MQPMR.

L'utilisation de cette option dispense l'application de la nécessité de réinitialiser la zone *MDMID* sur MINONE avant chaque appel MQPUT ou MQPUT1 .

### **PMNCID**

Générez un nouvel identificateur de corrélation.

Cette option permet au gestionnaire de files d'attente de remplacer le contenu de la zone *MDCID* dans MQMD par un nouvel identificateur de corrélation. Cet identificateur de corrélation est envoyé avec le message et renvoyé à l'application en sortie de l'appel MQPUT ou MQPUT1 .

Cette option peut également être spécifiée lorsque le message est inséré dans une liste de distribution. Pour plus de détails, voir la description de la zone *PRCID* dans la structure MQPMR.

PMNCID est utile dans les situations où l'application requiert un identificateur de corrélation unique.

**Options de groupe et de segment:** L'option suivante concerne le traitement des messages dans des groupes et des segments de messages logiques. Ces définitions peuvent vous aider à comprendre l'option:

### **Message physique**

Il s'agit de la plus petite unité d'informations pouvant être placée dans ou supprimée d'une file d'attente ; elle correspond souvent aux informations spécifiées ou extraites sur un seul appel MQPUT, MQPUT1 ou MQGET. Chaque message physique possède son propre descripteur de message (MQMD). En règle générale, les messages physiques se distinguent par des valeurs différentes pour l'identificateur de message (zone *MDMID* dans MQMD), bien que cela ne soit pas appliqué par le gestionnaire de files d'attente.

### **Message logique**

Il s'agit d'une unité unique d'informations d'application. En l'absence de contraintes système, un message logique serait identique à un message physique. Cependant, lorsque les messages logiques sont volumineux, les contraintes du système peuvent rendre souhaitable ou nécessaire la division d'un message logique en deux ou plusieurs messages physiques, appelés *segments*.

Un message logique segmenté se compose d'au moins deux messages physiques ayant le même identificateur de groupe non null (zone *MDGID* dans MQMD) et le même numéro de séquence de



message (zone *MDSEQ* dans *MQMD*). Les segments se distinguent par des valeurs différentes pour le décalage de segment (zone *MDOFF* dans *MQMD*), qui fournit le décalage des données dans le message physique à partir du début des données dans le message logique. Chaque segment étant un message physique, les segments d'un message logique ont généralement des identificateurs de message différents.

Un message logique qui n'a pas été segmenté, mais pour lequel la segmentation a été autorisée par l'application émettrice, possède également un identificateur de groupe non null, bien que dans ce cas, il n'y ait qu'un seul message physique avec cet identificateur de groupe si le message logique n'appartient pas à un groupe de messages. Les messages logiques pour lesquels la segmentation a été inhibée par l'application émettrice ont un identificateur de groupe nul (*GINONE*), sauf si le message logique appartient à un groupe de messages.

### **Groupe de messages**

Il s'agit d'un ensemble d'un ou de plusieurs messages logiques ayant le même identificateur de groupe non null. Les messages logiques du groupe se distinguent par des valeurs différentes pour le numéro de séquence de message, qui est un entier compris entre 1 et n, où n est le nombre de messages logiques du groupe. Si un ou plusieurs des messages logiques sont segmentés, il y a plus de n messages physiques dans le groupe.

### **PMLOGO**

Les messages des groupes et des segments de messages logiques sont placés dans l'ordre logique.

Cette option indique au gestionnaire de files d'attente comment l'application insère des messages dans des groupes et des segments de messages logiques. Elle ne peut être spécifiée que dans l'appel *MQPUT* ; elle n'est pas valide dans l'appel *MQPUT1* .

Si *PMLOGO* est spécifié, cela indique que l'application utilise des appels *MQPUT* successifs à :

- Placer les segments dans chaque message logique dans l'ordre croissant des décalages de segment, à partir de 0, sans écart.
- Placez tous les segments dans un message logique avant d'insérer les segments dans le message logique suivant.
- Placer les messages logiques dans chaque groupe de messages dans l'ordre croissant des numéros de séquence de message, à partir de 1, sans écart.
- Placez tous les messages logiques dans un groupe de messages avant de placer les messages logiques dans le groupe de messages suivant.

Cet ordre est appelé "ordre logique".

Etant donné que l'application a indiqué au gestionnaire de files d'attente comment elle insère des messages dans des groupes et des segments de messages logiques, l'application n'a pas besoin de gérer et de mettre à jour les informations de groupe et de segment relatives à chaque appel *MQPUT*, comme le fait le gestionnaire de files d'attente. En particulier, cela signifie que l'application n'a pas besoin de définir les zones *MDGID*, *MDSEQ* et *MDOFF* dans *MQMD*, car le gestionnaire de files d'attente les définit sur les valeurs appropriées. L'application a besoin de définir uniquement la zone *MDMFL* dans *MQMD*, pour indiquer quand les messages appartiennent à des groupes ou sont des segments de messages logiques, et pour indiquer le dernier message dans un groupe ou le dernier segment d'un message logique.

Une fois qu'un groupe de messages ou un message logique a été démarré, les appels *MQPUT* suivants doivent spécifier les indicateurs MF\* appropriés dans *MDMFL* dans *MQMD*. Si l'application tente d'insérer un message qui ne se trouve pas dans un groupe lorsqu'il existe un groupe de messages sans fin ou d'insérer un message qui n'est pas un segment lorsqu'il existe un message logique sans fin, l'appel échoue avec le code anomalie RC2241 ou RC2242 , selon le cas. Toutefois, le gestionnaire de files d'attente conserve les informations relatives au groupe de messages en cours ou au message logique en cours, et l'application peut les arrêter en envoyant un message (éventuellement sans données de message d'application) en spécifiant *MFLMIG* ou *MFLSEG*, selon le cas, avant d'émettre à nouveau l'appel *MQPUT* pour insérer le message qui n'est pas dans le groupe ou qui n'est pas un segment.

Le Tableau 716, à la page 1234 présente les combinaisons d'options et d'indicateurs valides, ainsi que les valeurs des zones *MDGID*, *MDSEQ* et *MDOFF* utilisées par le gestionnaire de files d'attente dans chaque cas. Les combinaisons d'options et d'indicateurs qui ne sont pas affichées dans le tableau ne sont pas valides. Les colonnes de la table ont les significations suivantes:

**NOM DE JOURNAL**

Indique si l'option *PMLOGO* est spécifiée lors de l'appel.

**MIG**

Indique si l'option *MFmig* ou *MFLmig* est indiquée sur l'appel.

**SEG**

Indique si l'option *MFseg* ou *MFLseg* est indiquée dans l'appel.

**SEG OK**

Indique si l'option *MFsega* est spécifiée dans l'appel.

**Groupe de travail en cours**

Indique si un groupe de messages en cours existe avant l'appel.

**Message du journal en cours**

Indique si un message logique en cours existe avant l'appel.

**Autres colonnes**

Affiche les valeurs utilisées par le gestionnaire de files d'attente. "Précédent" indique la valeur utilisée pour la zone du message précédent pour l'identificateur de file d'attente.

**PMRLOC**

Indique que le *PMRQN* de la structure *MQPMO* doit être terminé avec le nom de la file d'attente locale dans laquelle le message est réellement inséré. Le nom *ResolvedQMgrest* également complété par le nom du gestionnaire de files d'attente local qui héberge la file d'attente locale. Voir *ORLOQ* pour ce que cela signifie. Si un utilisateur est autorisé à effectuer une insertion dans une file d'attente, il dispose des droits requis pour spécifier cet indicateur dans l'appel *MQPUT*. Aucun droit spécial n'est requis.

Tableau 716. Options *MQPUT* relatives aux messages dans les groupes et les segments de messages logiques

Options que vous spécifiez				Statut du groupe et du message de journal avant l'appel		Valeurs utilisées par le gestionnaire de files d'attente		
NOM DE JOURNAL	MIG	SEG	SEG OK	Group e en cours	Messa ge du journa l en cours	<i>MDGID</i>	<i>MDSEQ</i>	<i>MDOFF</i>
Oui	Non	Non	Non	Non	Non	GINONE	1	0
Oui	Non	Non	Oui	Non	Non	Nouvel ID de groupe	1	0
Oui	Non	Oui	Oui ou Non	Non	Non	Nouvel ID de groupe	1	0
Oui	Non	Oui	Oui ou Non	Non	Oui	ID groupe précédent	1	Décalage précédent + longueur de segment précédent
Oui	Oui	Oui ou Non	Oui ou Non	Non	Non	Nouvel ID de groupe	1	0

Tableau 716. Options MQPUT relatives aux messages dans les groupes et les segments de messages logiques (suite)

Options que vous spécifiez				Statut du groupe et du message de journal avant l'appel		Valeurs utilisées par le gestionnaire de files d'attente		
Oui	Oui	Oui ou Non	Oui ou Non	Oui	Non	ID groupe précédent	Numéro de séquence précédent + 1	0
Oui	Oui	Oui	Oui ou Non	Oui	Oui	ID groupe précédent	Numéro de séquence précédent	Décalage précédent + longueur de segment précédent
Non	Non	Non	Non	Oui ou Non	Oui ou Non	GINONE	1	0
Non	Non	Non	Oui	Oui ou Non	Oui ou Non	Nouvel ID de groupe si GINONE, autre valeur dans la zone	1	0
Non	Non	Oui	Oui ou Non	Oui ou Non	Oui ou Non	Nouvel ID de groupe si GINONE, autre valeur dans la zone	1	Valeur dans la zone
Non	Oui	Non	Oui ou Non	Oui ou Non	Oui ou Non	Nouvel ID de groupe si GINONE, autre valeur dans la zone	Valeur dans la zone	0
Non	Oui	Oui	Oui ou Non	Oui ou Non	Oui ou Non	Nouvel ID de groupe si GINONE, autre valeur dans la zone	Valeur dans la zone	Valeur dans la zone

**Remarque :**

- PMLOGO n'est pas valide sur l'appel MQPUT1 .
- Pour la zone *MDMID* , le gestionnaire de files d'attente génère un nouvel identificateur de message si PMNMID ou MINONE est spécifié et utilise la valeur de la zone dans le cas contraire.
- Pour la zone *MDCID* , le gestionnaire de files d'attente génère un nouvel identificateur de corrélation si PMNCID est spécifié et utilise la valeur de la zone dans le cas contraire.

Lorsque PMLOGO est spécifié, le gestionnaire de files d'attente requiert que tous les messages d'un groupe et les segments d'un message logique soient insérés avec la même valeur dans la zone *MDPER* de MQMD, c'est-à-dire que tous les messages soient persistants ou non persistants. Si cette condition n'est pas satisfaite, l'appel MQPUT échoue avec le code anomalie RC2185 .

L'option PMLOGO affecte les unités de travail comme suit:

- Si le premier message physique d'un groupe ou d'un message logique est inséré dans une unité d'oeuvre, tous les autres messages physiques du groupe ou du message logique doivent être insérés dans une unité d'oeuvre, si le même descripteur de file d'attente est utilisé. Toutefois, ils ne doivent pas être placés dans la même unité de travail. Cela permet de diviser un groupe

de messages ou un message logique composé de plusieurs messages physiques en plusieurs unités de travail consécutives pour l'identificateur de file d'attente.

- Si le premier message physique d'un groupe ou d'un message logique n'est pas inséré dans une unité d'oeuvre, aucun des autres messages physiques du groupe ou du message logique ne peut être inséré dans une unité d'oeuvre si le même descripteur de file d'attente est utilisé.

Si ces conditions ne sont pas satisfaites, l'appel MQPUT échoue avec le code anomalie RC2245 .

Lorsque PMLOGO est spécifié, le MQMD fourni dans l'appel MQPUT ne doit pas être inférieur à MDVER2. Si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie RC2257 .

Si PMLOGO n'est pas spécifié, les messages des groupes et des segments de messages logiques peuvent être insérés dans n'importe quel ordre et il n'est pas nécessaire d'insérer des groupes de messages complets ou des messages logiques complets. Il incombe à l'application de s'assurer que les zones *MDGID*, *MDSEQ*, *MDOFF* et *MDMFL* ont les valeurs appropriées.

Il s'agit de la technique qui peut être utilisée pour redémarrer un groupe de messages ou un message logique au milieu, après une défaillance du système. Lorsque le système redémarre, l'application peut définir les zones *MDGID*, *MDSEQ*, *MDOFF*, *MDMFL* et *MDPER* sur les valeurs appropriées, puis émettre l'appel MQPUT avec PMSYP ou PMNSYP défini comme *nécessaire*, mais sans spécifier PMLOGO. Si cet appel aboutit, le gestionnaire de files d'attente conserve les informations de groupe et de segment, et les appels MQPUT ultérieurs utilisant cet identificateur de file d'attente peuvent spécifier PMLOGO comme normal.

Les informations de groupe et de segment que le gestionnaire de files d'attente conserve pour l'appel MQPUT sont distinctes des informations de groupe et de segment qu'il conserve pour l'appel MQGET.

Pour un descripteur de file d'attente donné, l'application est libre de mélanger les appels MQPUT qui spécifient PMLOGO avec les appels MQPUT qui ne le font pas, mais les points suivants doivent être notés:

- Si PMLOGO n'est pas spécifié, chaque appel MQPUT réussi entraîne le gestionnaire de files d'attente à définir les informations de groupe et de segment pour l'identificateur de file d'attente sur les valeurs spécifiées par l'application ; cela remplace les informations de groupe et de segment existantes conservées par le gestionnaire de files d'attente pour l'identificateur de file d'attente.
- Si PMLOGO n'est pas spécifié, l'appel n'échoue pas s'il existe un groupe de messages ou un message logique en cours ; l'appel peut toutefois aboutir avec un code achèvement CCWARN. Le Tableau 717, à la page 1236 montre les différentes observations qui peuvent survenir. Dans ces cas, si le code achèvement n'est pas CCOK, le code raison est l'un des suivants (selon le cas):
  - RC2241
  - RC2242
  - RC2185
  - RC2245

**Remarque :** Le gestionnaire de files d'attente ne vérifie pas les informations de groupe et de segment pour l'appel MQPUT1 .

<i>Tableau 717. Résultat lorsque l'appel MQPUT ou MQCLOSE n'est pas cohérent avec les informations de groupe et de segment</i>		
<b>L'appel en cours est</b>	<b>L'appel précédent était MQPUT avec PMLOGO</b>	<b>L'appel précédent était MQPUT sans PMLOGO</b>
MQPUT avec PMLOGO	CCFAIL	CCFAIL
MQPUT sans PMLOGO	CCWARN	CCOK

Tableau 717. Résultat lorsque l'appel MQPUT ou MQCLOSE n'est pas cohérent avec les informations de groupe et de segment (suite)		
L'appel en cours est	L'appel précédent était MQPUT avec PMLOGO	L'appel précédent était MQPUT sans PMLOGO
MQCLOSE avec un groupe non terminé ou un message logique	CCWARN	CCOK

Les applications qui souhaitent simplement placer des messages et des segments dans l'ordre logique sont recommandées pour spécifier PMLOGO, car il s'agit de l'option la plus simple à utiliser. Cette option dispense l'application de la nécessité de gérer les informations de groupe et de segment, car le gestionnaire de files d'attente gère ces informations. Toutefois, les applications spécialisées peuvent avoir besoin d'un contrôle plus important que celui fourni par l'option PMLOGO, et cela peut être réalisé en ne spécifiant pas cette option. Dans ce cas, l'application doit s'assurer que les zones *MDGID*, *MDSEQ*, *MDOFF* et *MDMFL* de MQMD sont définies correctement avant chaque appel MQPUT ou MQPUT1.

Par exemple, une application qui souhaite transmettre des messages physiques qu'elle reçoit, indépendamment du fait que ces messages se trouvent dans des groupes ou des segments de messages logiques, ne doit pas spécifier PMLOGO. Il y a deux raisons à cela:

- Si les messages sont extraits et placés dans l'ordre, la spécification de PMLOGO entraîne l'affectation d'un nouvel identificateur de groupe aux messages, ce qui peut rendre difficile ou impossible pour l'émetteur des messages de corréler les messages de réponse ou de rapport qui résultent du groupe de messages.
- Dans un réseau complexe avec plusieurs chemins entre les gestionnaires de files d'attente d'envoi et de réception, les messages physiques peuvent arriver dans le désordre. En ne spécifiant pas PMLOGO et le GMLOGO correspondant sur l'appel MQGET, l'application de réacheminement peut extraire et réacheminer chaque message physique dès qu'il arrive, sans avoir à attendre le message suivant dans l'ordre logique d'arrivée.

Les applications qui génèrent des messages de rapport pour des messages dans des groupes ou des segments de messages logiques ne doivent pas non plus spécifier PMLOGO lors de l'insertion du message de rapport.

PMLOGO peut être spécifié avec l'une des autres options PM\*.

**Options de contexte:** Les options suivantes contrôlent le traitement du contexte de message:

#### PMNOC

Aucun contexte ne doit être associé au message.

Le contexte d'identité et le contexte d'origine sont définis pour indiquer l'absence de contexte. Cela signifie que les zones de contexte dans MQMD sont définies sur:

- Blancs pour les zones de caractères
- Valeurs nulles pour les zones d'octet
- Zéros pour les champs numériques

#### PMDEFC

Utiliser le contexte par défaut.

Le message doit être associé à des informations de contexte par défaut, à la fois pour l'identité et l'origine. Le gestionnaire de files d'attente définit les zones de contexte dans le descripteur de message comme suit:

Tableau 718. Valeurs d'informations contextuelles par défaut pour les zones MQMD

Zone dans MQMD	Valeur utilisée
MDUID	Déterminé à partir de l'environnement si possible ; sinon, il est mis à blanc.
MDACC	Déterminé à partir de l'environnement si possible ; défini sur ACNONE dans le cas contraire.
MDAID	Mettez à blanc.
MDPAT	Déterminé à partir de l'environnement.
MDPAN	Déterminé à partir de l'environnement si possible ; sinon, il est mis à blanc.
MDPD	Définir la date à laquelle le message est inséré.
MDPT	Défini sur l'heure à laquelle le message est inséré.
MDAOD	Mettez à blanc.

Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

Il s'agit de l'action par défaut si aucune option de contexte n'est spécifiée.

#### **PMPASI**

Transmettez le contexte d'identité à partir d'un descripteur de file d'attente d'entrée.

Le message doit être associé à des informations de contexte. Le contexte d'identité est extrait de l'identificateur de file d'attente spécifié dans la zone *PMCT*. Les informations de contexte d'origine sont générées par le gestionnaire de files d'attente de la même manière que pour *PMDEFC* (voir le tableau précédent pour les valeurs). Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

Pour l'appel *MQPUT*, la file d'attente doit avoir été ouverte avec l'option *OOPASI* (ou une option qui l'implique). Pour l'appel *MQPUT1*, la même vérification d'autorisation est effectuée que pour l'appel *MQOPEN* avec l'option *OOPASI*.

#### **PMPASA**

Transmettez tout le contexte à partir d'un descripteur de file d'attente d'entrée.

Le message doit être associé à des informations de contexte. L'identité et le contexte d'origine sont extraits du descripteur de file d'attente spécifié dans la zone *PMCT*. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

Pour l'appel *MQPUT*, la file d'attente doit avoir été ouverte avec l'option *OOPASA* (ou une option qui l'implique). Pour l'appel *MQPUT1*, la même vérification d'autorisation est effectuée que pour l'appel *MQOPEN* avec l'option *OOPASA*.

#### **PMSETI**

Définissez le contexte d'identité à partir de l'application.

Le message doit être associé à des informations de contexte. L'application spécifie le contexte d'identité dans la structure *MQMD*. Les informations de contexte d'origine sont générées par le gestionnaire de files d'attente de la même manière que pour *PMDEFC* (voir le tableau précédent pour les valeurs). Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

Pour l'appel *MQPUT*, la file d'attente doit avoir été ouverte avec l'option *OOSETI* (ou une option qui l'implique). Pour l'appel *MQPUT1*, la même vérification d'autorisation est effectuée que pour l'appel *MQOPEN* avec l'option *OOSETI*.

#### **PMSETA**

Définissez tous les contextes à partir de l'application.

Le message doit être associé à des informations de contexte. L'application spécifie le contexte d'identité et d'origine dans la structure MQMD. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

Pour l'appel MQPUT, la file d'attente doit avoir été ouverte avec l'option OOSETA. Pour l'appel MQPUT1, la même vérification d'autorisation est effectuée que pour l'appel MQOPEN avec l'option OOSETA.

Une seule des options de contexte PM\* peut être spécifiée. Si aucune de ces options n'est spécifiée, PMDEFC est utilisé.

**Types de réponse d'insertion.** Les options suivantes contrôlent la réponse renvoyée à un appel MQPUT ou MQPUT1. Vous ne pouvez spécifier qu'une seule de ces options. Si PMARES et PMSRES ne sont pas indiqués, PMRASQ ou PMRAST est utilisé par défaut.

### **PMARES**

L'option PMARES demande qu'une opération MQPUT ou MQPUT1 soit exécutée sans que l'application attende que le gestionnaire de files d'attente termine l'appel. L'utilisation de cette option permet d'améliorer les performances de la messagerie, en particulier pour les applications utilisant des liaisons client. Une application peut vérifier périodiquement, à l'aide de l'instruction MQSTAT, si une erreur s'est produite lors des appels asynchrones précédents.

Avec cette option, seules les zones suivantes sont garanties pour être renseignées dans le MQMD ;

- MDAID
- MDPAT
- MDPAN
- MDAOD

De plus, si l'un ou l'autre des éléments PMNMID ou PMNCID ou les deux sont spécifiés en tant qu'options, les éléments MDMID et MDCID renvoyés sont également terminés. (PMNMID peut être implicitement spécifié en spécifiant une zone MDMID vide).

Seules les zones précédemment indiquées sont renseignées. D'autres informations normalement renvoyées dans la structure MQMD ou MQPMO ne sont pas définies.

Lorsque vous demandez une réponse d'insertion asynchrone pour MQPUT ou MQPUT1, un CMPCOD et un RAISON de CCOK et RCNONE ne signifient pas nécessairement que le message a été correctement inséré dans une file d'attente. Lors du développement d'une application MQI qui utilise une réponse d'insertion asynchrone et qui requiert la confirmation que des messages ont été insérés dans une file d'attente, vous devez vérifier les codes CMPCOD et REASON des opérations d'insertion et utiliser MQSTAT pour demander des informations sur les erreurs asynchrones.

Bien que la réussite ou l'échec de chaque appel MQPUT/MQPUT1 individuel puisse ne pas être renvoyé immédiatement, la première erreur qui s'est produite lors d'un appel asynchrone peut être déterminée ultérieurement via un appel à MQSTAT.

Si un message persistant sous le point de synchronisation ne parvient pas à être distribué à l'aide d'une réponse d'insertion asynchrone et que vous tentez de valider la transaction, la validation échoue et la transaction est annulée avec le code achèvement CCFAIL et le motif RC2003.

L'application peut effectuer un appel à MQSTAT pour déterminer la cause d'un échec MQPUT ou MQPUT1 précédent.

### **PMSRES**

La spécification de cette valeur pour une option d'insertion dans la structure MQPMO garantit que l'opération MQPUT ou MQPUT1 est toujours émise de manière synchrone. Si l'opération aboutit, toutes les zones de MQMD et MQPMO sont terminées. Il est fourni pour garantir une réponse synchrone quelle que soit la valeur de réponse d'insertion par défaut définie dans la file d'attente ou l'objet de rubrique.

### **PMRASQ**

Si cette valeur est spécifiée pour un appel MQPUT, le type de réponse d'insertion utilisé est extrait de la valeur DEFRESP spécifiée dans la file d'attente lors de son ouverture par l'application. Si

une application client est connectée à un gestionnaire de files d'attente à un niveau antérieur à IBM WebSphere MQ 7.0, elle se comporte comme si PMSRES était spécifié.

Si cette option est spécifiée pour un appel MQPUT1, la valeur DEFRESP de la définition de file d'attente n'est pas utilisée. Si l'appel MQPUT1 utilise PMSYP, il se comporte comme pour PMARES, et s'il utilise PMNSYP, il se comporte comme pour PMSRES.

#### **PMRAST**

Il s'agit d'un synonyme de PMRASQ à utiliser avec les objets de rubrique.

**Autres options:** Les options suivantes contrôlent la vérification des autorisations et ce qui se passe lorsque le gestionnaire de files d'attente est mis au repos:

#### **PMALTU**

Valider avec l'ID utilisateur spécifié.

Cela indique que la zone *ODAU* du paramètre **OBJDSC** de l'appel MQPUT1 contient un identificateur utilisateur qui doit être utilisé pour valider les droits d'accès permettant d'insérer des messages dans la file d'attente. L'appel peut aboutir uniquement si *ODAU* est autorisé à ouvrir la file d'attente avec les options spécifiées, que l'ID utilisateur sous lequel l'application est exécutée soit autorisé ou non à le faire. (Cela ne s'applique pas aux options de contexte spécifiées, qui sont toujours vérifiées par rapport à l'identificateur utilisateur sous lequel l'application s'exécute.)

Cette option est valide uniquement avec l'appel MQPUT1.

#### **PMFIQ**

Echec si le gestionnaire de files d'attente est mis au repos.

Cette option force l'échec de l'appel MQPUT ou MQPUT1 si le gestionnaire de files d'attente est à l'état de mise au repos.

L'appel renvoie le code achèvement CCFAIL avec le code anomalie RC2161.

**Option par défaut:** si aucune des options décrites précédemment n'est requise, l'option suivante peut être utilisée:

#### **PMAUCUN**

Aucune option spécifiée.

Cette valeur peut être utilisée pour indiquer qu'aucune autre option n'a été spécifiée ; toutes les options prennent leurs valeurs par défaut. PMNONE est défini pour faciliter la documentation du programme ; il n'est pas prévu que cette option soit utilisée avec d'autres, mais comme sa valeur est égale à zéro, une telle utilisation ne peut pas être détectée.

Il s'agit d'une zone d'entrée. La valeur initiale de la zone *PMOPT* est PMNONE.

#### **PMPRF (entier signé à 10 chiffres)**

Indicateurs indiquant les zones MQPMR qui sont présentes.

Cette zone contient des indicateurs qui doivent être définis pour indiquer les zones MQPMR présentes dans les enregistrements de message d'insertion fournis par l'application. *PMPRF* est utilisé uniquement lorsque le message est inséré dans une liste de distribution. La zone est ignorée si *PMREC* est égal à zéro, ou si les deux valeurs *PMPRO* et *PMPRP* sont égales à zéro.

Pour les zones qui sont présentes, le gestionnaire de files d'attente utilise pour chaque destination les valeurs des zones de l'enregistrement de message d'insertion correspondant. Pour les zones absentes, le gestionnaire de files d'attente utilise les valeurs de la structure MQMD.

Un ou plusieurs des indicateurs suivants peuvent être spécifiés pour indiquer les zones présentes dans les enregistrements de message d'insertion:

#### **PFMID**

La zone d'identificateur de message est présente.

#### **ID de fonction**

La zone d'identificateur de corrélation est présente.



**ID de groupe de travail**

La zone d'identificateur de groupe est présente.

**PFFB**

Un champ de commentaires est présent.

**PFACC**

La zone de jeton de comptabilité est présente.

Si cet indicateur est spécifié, PMSETI ou PMSETA doit être spécifié dans la zone *PMOPT* ; si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie RC2158 .

Si aucune zone MQPMR n'est présente, les éléments suivants peuvent être spécifiés:

**PFNONE**

Aucune zone d'enregistrement d'insertion de message n'est présente.

Si cette valeur est spécifiée, soit *PMREC* doit être égal à zéro, soit les deux valeurs *PMPRO* et *PMPRP* doivent être égales à zéro.

PFNONE est défini pour aider à la documentation du programme. Il n'est pas prévu que cette constante soit utilisée avec une autre, mais comme sa valeur est nulle, une telle utilisation ne peut pas être détectée.

Si *PMPRF* contient des indicateurs qui ne sont pas valides ou que des enregistrements de message d'insertion sont fournis mais que *PMPRF* a la valeur PFNONE, l'appel échoue avec le code anomalie RC2158 .

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est PFNONE. Cette zone est ignorée si *PMVER* est inférieur à *PMVER2*.

**PMPRO (entier signé à 10 chiffres)**

Décalage du premier enregistrement de message d'insertion à partir du début de MQPMO.

Décalage en octets du premier enregistrement de message d'insertion MQPMR à partir du début de la structure MQPMO. Le décalage peut être positif ou négatif. *PMPRO* est utilisé uniquement lorsque le message est inséré dans une liste de distribution. La zone est ignorée si *PMREC* a la valeur zéro.

Lorsque le message est inséré dans une liste de distribution, un tableau d'un ou de plusieurs enregistrements de message d'insertion MQPMR peut être fourni afin de spécifier certaines propriétés du message pour chaque destination individuellement ; ces propriétés sont les suivantes:

- identificateur de message
- identificateur de corrélation
- identificateur de groupe
- valeur de retour
- jeton de comptabilité

Il n'est pas nécessaire de spécifier toutes ces propriétés, mais quel que soit le sous-ensemble choisi, les zones doivent être spécifiées dans le bon ordre. Pour plus de détails, voir la description de la structure MQPMR.

En général, il doit y avoir autant d'enregistrements de message d'insertion que d'enregistrements d'objet spécifiés par MQOD lors de l'ouverture de la liste de distribution ; chaque enregistrement de message d'insertion fournit les propriétés de message pour la file d'attente identifiée par l'enregistrement d'objet correspondant. Les files d'attente de la liste de distribution dont l'ouverture a échoué doivent toujours avoir des enregistrements de message d'insertion alloués aux emplacements appropriés dans le tableau, bien que les propriétés de message soient ignorées dans ce cas.

Il est possible que le nombre d'enregistrements de message d'insertion soit différent du nombre d'enregistrements d'objet. S'il y a moins d'enregistrements de message d'insertion que d'enregistrements d'objet, les propriétés de message pour les destinations qui n'ont pas d'enregistrements de message d'insertion sont extraites des zones correspondantes dans le descripteur de message MQMD. S'il y a plus d'enregistrements de message d'insertion que

d'enregistrements d'objet, l'excès n'est pas utilisé (bien qu'il soit toujours possible d'y accéder). Les enregistrements de message d'insertion sont facultatifs, mais s'ils sont fournis, ils doivent être *PMREC*.

Les enregistrements de message d'insertion peuvent être fournis de la même manière que les enregistrements d'objet dans MQOD, soit en spécifiant un décalage dans *PMPRO*, soit en spécifiant une adresse dans *PMPRP*; Pour plus d'informations sur cette opération, voir la zone *ODORO* décrite dans «MQOD (descripteur d'objet) sous IBM i», à la page 1215.

Vous ne pouvez pas utiliser plus de l'une des valeurs *PMPRO* et *PMPRP*; l'appel échoue avec le code anomalie RC2159 si les deux valeurs sont différentes de zéro.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *PMVER* est inférieur à *PMVER2*.

### **PMPRP (pointeur)**

Adresse du premier enregistrement de message d'insertion.

Il s'agit de l'adresse du premier enregistrement de message d'insertion MQPMR. *PMPRP* est utilisé uniquement lorsque le message est inséré dans une liste de distribution. La zone est ignorée si *PMREC* a la valeur zéro.

*PMPRP* ou *PMPRO* peut être utilisé pour spécifier les enregistrements de message d'insertion, mais pas les deux; voir la description de la zone *PMRRO* pour plus de détails. Si *PMPRP* n'est pas utilisé, il doit être défini sur le pointeur null ou sur les octets null.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null. Cette zone est ignorée si *PMVER* est inférieur à *PMVER2*.

### **PMREC (entier signé à 10 chiffres)**

Nombre d'enregistrements de message d'insertion ou d'enregistrements de réponse présents.

Il s'agit du nombre d'enregistrements de message d'insertion MQPMR ou d'enregistrements de réponse MQRR fournis par l'application. Ce nombre peut être supérieur à zéro uniquement si le message est inséré dans une liste de distribution. Les enregistrements de message d'insertion et les enregistrements de réponse sont facultatifs-l'application n'a pas besoin de fournir d'enregistrements, ou elle peut choisir de fournir des enregistrements d'un seul type. Toutefois, si l'application fournit des enregistrements des deux types, elle doit fournir des enregistrements *PMREC* de chaque type.

Il n'est pas nécessaire que la valeur de *PMREC* soit identique au nombre de destinations dans la liste de distribution. Si trop d'enregistrements sont fournis, l'excès n'est pas utilisé; si trop peu d'enregistrements sont fournis, des valeurs par défaut sont utilisées pour les propriétés de message pour les destinations qui n'ont pas d'enregistrements de message d'insertion (voir *PMPRO* plus loin dans cette rubrique).

Si *PMREC* est inférieur à zéro ou supérieur à zéro mais que le message n'est pas inséré dans une liste de distribution, l'appel échoue avec le code anomalie RC2154.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *PMVER* est inférieur à *PMVER2*.

### **PMRMN (chaîne de caractères de 48 octets)**

Nom du gestionnaire de files d'attente de destination résolu.

Il s'agit du nom du gestionnaire de files d'attente de destination une fois la résolution de nom effectuée par le gestionnaire de files d'attente local. Le nom renvoyé correspond au nom du gestionnaire de files d'attente qui possède la file d'attente identifiée par *PMRQNet* peut correspondre au nom du gestionnaire de files d'attente local.

Si *PMRQN* est une file d'attente partagée appartenant au groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local, *PMRMN* est le nom du groupe de partage de files d'attente. Si la file d'attente appartient à un autre groupe de partage de files d'attente, *PMRQN* peut être le nom du groupe de partage de files d'attente ou le nom d'un gestionnaire de files d'attente

membre du groupe de partage de files d'attente (la nature de la valeur renvoyée est déterminée par les définitions de files d'attente qui existent sur le gestionnaire de files d'attente local).

Une valeur non vide est renvoyée uniquement si l'objet est une file d'attente unique ; si l'objet est une liste de distribution ou une rubrique, la valeur renvoyée n'est pas définie.

Il s'agit d'une zone de sortie. La longueur de cette zone est indiquée par LNQM. La valeur initiale de cette zone est de 48 caractères blancs.

### **PMRQ (chaîne de caractères de 48 octets)**

Nom résolu de la file d'attente de destination.

Il s'agit du nom de la file d'attente de destination une fois la résolution de nom effectuée par le gestionnaire de files d'attente local. Le nom renvoyé est le nom d'une file d'attente qui existe sur le gestionnaire de files d'attente identifié par *PMRMN*.

Une valeur non vide est renvoyée uniquement si l'objet est une file d'attente unique ; si l'objet est une liste de distribution ou une rubrique, la valeur renvoyée n'est pas définie.

Il s'agit d'une zone de sortie. La longueur de cette zone est indiquée par LNQN. La valeur initiale de cette zone est de 48 caractères blancs.

### **PMRRO (entier signé à 10 chiffres)**

Décalage du premier enregistrement de réponse à partir du début de MQPMO.

Décalage en octets du premier enregistrement de réponse MQRR à partir du début de la structure MQPMO. Le décalage peut être positif ou négatif. *PMRRO* est utilisé uniquement lorsque le message est inséré dans une liste de distribution. La zone est ignorée si *PMREC* a la valeur zéro.

Lorsque le message est inséré dans une liste de distribution, un tableau d'un ou de plusieurs enregistrements de réponse MQRR peut être fourni afin d'identifier les files d'attente auxquelles le message n'a pas été envoyé correctement (zone *RRCC* dans MQRR), ainsi que la raison de chaque échec (zone *RRREA* dans MQRR). Il se peut que le message n'ait pas été envoyé en raison de l'échec de l'ouverture de la file d'attente ou de l'échec de l'opération d'insertion. Le gestionnaire de files d'attente définit les enregistrements de réponse uniquement lorsque le résultat de l'appel est mixte (c'est-à-dire que certains messages ont été envoyés avec succès alors que d'autres ont échoué ou que tous ont échoué, mais pour des raisons différentes) ; le code anomalie RC2136 de l'appel indique ce cas. Si le même code anomalie s'applique à toutes les files d'attente, ce code anomalie est renvoyé dans le paramètre **REASON** de l'appel MQPUT ou MQPUT1 et les enregistrements de réponse ne sont pas définis.

En général, il doit y avoir autant d'enregistrements de réponse que d'enregistrements d'objet spécifiés par MQOD lors de l'ouverture de la liste de distribution ; si nécessaire, chaque enregistrement de réponse est défini sur le code achèvement et le code raison de l'insertion dans la file d'attente identifiée par l'enregistrement d'objet correspondant. Les files d'attente de la liste de distribution dont l'ouverture a échoué doivent toujours avoir des enregistrements de réponse alloués aux positions appropriées dans le tableau, bien qu'elles soient définies sur le code achèvement et le code raison résultant de l'opération d'ouverture, plutôt que sur l'opération d'insertion.

Il est possible que le nombre d'enregistrements de réponse soit différent du nombre d'enregistrements d'objet. S'il y a moins d'enregistrements de réponse que d'enregistrements d'objet, il se peut que l'application ne puisse pas identifier toutes les destinations pour lesquelles l'opération d'insertion a échoué ou les raisons des échecs. S'il y a plus d'enregistrements de réponse que d'enregistrements d'objet, l'excédent n'est pas utilisé (bien qu'il soit toujours possible d'y accéder). Les enregistrements de réponse sont facultatifs, mais s'ils sont fournis, ils doivent être *PMREC* .

Les enregistrements de réponse peuvent être fournis de la même manière que les enregistrements d'objet dans MQOD, soit en spécifiant un décalage dans *PMRRO*, soit en spécifiant une adresse dans *PMRRP* ; Pour plus d'informations sur cette opération, voir la zone *ODORO* décrite dans «MQOD (descripteur d'objet) sous IBM i», à la page 1215. Toutefois, vous ne pouvez pas utiliser plus d'un *PMRRO* ou *PMRRP* ; l'appel échoue avec le code anomalie RC2156 si les deux sont différents de zéro.

Pour l'appel MQPUT1 , cette zone doit être à zéro. En effet, les informations de réponse (si elles sont demandées) sont renvoyées dans les enregistrements de réponse spécifiés par le descripteur d'objet MQOD.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *PMVER* est inférieur à *PMVER2*.

### **PMRRP (pointeur)**

Adresse du premier enregistrement de réponse.

Il s'agit de l'adresse du premier enregistrement de réponse MQRR. *PMRRP* est utilisé uniquement lorsque le message est inséré dans une liste de distribution. La zone est ignorée si *PMREC* a la valeur zéro.

*PMRRP* ou *PMRRO* peut être utilisé pour spécifier les enregistrements de réponse, mais pas les deux ; voir la description de la zone *PMRRO* pour plus de détails. Si *PMRRP* n'est pas utilisé, il doit être défini sur le pointeur null ou sur les octets null.

Pour l'appel MQPUT1 , cette zone doit être le pointeur null ou des octets null. En effet, les informations de réponse (si elles sont demandées) sont renvoyées dans les enregistrements de réponse spécifiés par le descripteur d'objet MQOD.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null. Cette zone est ignorée si *PMVER* est inférieur à *PMVER2*.

### **PMSID (chaîne de caractères de 4 octets)**

Identificateur de structure.

La valeur doit être:

#### **PMSIDV**

Identificateur de la structure des options d'insertion de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est PMSIDV.

### **PMSL (MQLONG)**

Niveau d'abonnement ciblé par cette publication.

Seuls les abonnements dont le *PMSL* le plus élevé est inférieur ou égal à cette valeur reçoivent cette publication. Cette valeur doit être comprise entre zéro et 9 ; zéro est le niveau le plus bas.

La valeur initiale de cette zone est 9.

### **PMTO (entier signé à 10 chiffres)**

Réservé.

Il s'agit d'une zone réservée ; sa valeur n'est pas significative. La valeur initiale de cette zone est -1.

### **PMUDC (entier signé à 10 chiffres)**

Nombre de messages envoyés avec succès aux files d'attente éloignées.

Il s'agit du nombre de messages que l'appel MQPUT ou MQPUT1 en cours a correctement envoyés aux files d'attente de la liste de distribution qui sont résolues en files d'attente éloignées. Messages que le gestionnaire de files d'attente conserve temporairement sous forme de liste de distribution, en tant que nombre de destinations individuelles que ces listes de distribution contiennent. Cette zone est également définie lors de l'insertion d'un message dans une file d'attente unique qui ne figure pas dans une liste de distribution.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0. Cette zone n'est pas définie si *PMVER* est inférieur à *PMVER2*.

### **PMVER (entier signé à 10 chiffres)**

Numéro de version de la structure.

Il doit s'agir de l'une des valeurs suivantes :

**PMVER1**

Structure des options d'insertion de message Version-1 .

**PMVER2**

Structure des options d'insertion de message Version-2 .

Les zones qui existent uniquement dans la version la plus récente de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

**PMVERC**

Version actuelle de la structure des options d'insertion de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est PMVER1.

**Valeurs initiales**

Tableau 719. Zones dans MQPMO		
Nom de zone	Nom de la constante	Valeur de la constante
PMSID	PMSIDV	'PMO~'
PMVER	PMVER1	1
PMOPT	PMAUCUN	0
PMTO	Aucun	-1
PMCT	Aucun	0
PMKDC	Aucun	0
PMUDC	Aucun	0
PMIDC	Aucun	0
PMRQN	Aucun	Espaces vides
PMRMN	Aucun	Espaces vides
PMREC	Aucun	0
PMPRF	PFNONE	0
PMPRO	Aucun	0
PMRRO	Aucun	0
PMPRP	Aucun	Pointeur null ou octets null
PMRRP	Aucun	Pointeur null ou octets null
<b>Remarque :</b>		
1. Le symbole ~ représente un caractère blanc unique.		

**Déclaration RPG**

```

D*..1.....2.....3.....4.....5.....6.....7..
D* MQPMO Structure
D*
D* Structure identifier
D PMSID          1      4   INZ('PMO ')
D* Structure version number
D PMVER          5      8I 0 INZ(1)
D* Options that control the action of MQPUT and MQPUT1
D PMOPT          9      12I 0 INZ(0)
D* Reserved

```

```

D PMTO                13      16I 0 INZ(-1)
D* Object handle of input queue
D PMCT                17      20I 0 INZ(0)
D* Number of messages sent successfully to local queues
D PMKDC               21      24I 0 INZ(0)
D* Number of messages sent successfully to remote queues
D PMUDC               25      28I 0 INZ(0)
D* Number of messages that could not be sent
D PMIDC               29      32I 0 INZ(0)
D* Resolved name of destination queue
D PMRQN               33      80      INZ
D* Resolved name of destination queue manager
D PMRMN               81      128     INZ
D* Number of put message records or response records present
D PMREC               129     132I 0 INZ(0)
D* Flags indicating which MQPMR fields are present
D PMPRF               133     136I 0 INZ(0)
D* Offset of first put message record from start of MQPMO
D PMPRO               137     140I 0 INZ(0)
D* Offset of first response record from start of MQPMO
D PMRRO               141     144I 0 INZ(0)
D* Address of first put message record
D PMPRP               145     160*    INZ(*NULL)
D* Address of first response record
D PMRRP               161     176*    INZ(*NULL)
D* Original message handle
D PMOMH               177     184I 0
D* New message handle
D PMNMH               185     190I 0
D* The action being performed
D PMACT               191     194I 0
D* Reserved
D PMRE1               195     198I 0

```

## IBM i MQPMR (enregistrement de message d'insertion) sur IBM i

La structure MQPMR permet de spécifier diverses propriétés de message pour une destination unique lorsqu'un message est inséré dans une liste de distribution.

### Présentation

**Objectif:** MQPMR est une structure d'entrée/sortie pour les appels MQPUT et MQPUT1 .

**Jeu de caractères et codage:** les données dans MQPMR doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par ENNAT. Toutefois, si l'application s'exécute en tant que client IBM MQ , la structure doit être dans le jeu de caractères et le codage du client.

**Syntaxe:** en fournissant un tableau de ces structures sur l'appel MQPUT ou MQPUT1 , il est possible de spécifier des valeurs différentes pour chaque file d'attente de destination dans une liste de distribution. Certaines des zones sont des entrées uniquement, d'autres sont des entrées / sorties.

**Remarque :** Cette structure est inhabituelle dans la mesure où elle n'a pas de présentation fixe. Les zones de cette structure sont facultatives et la présence ou l'absence de chaque zone est indiquée par les indicateurs de la zone *PMPRF* dans MQPMO. Les zones qui sont présentes **doivent apparaître dans l'ordre suivant :**

- *PRMID*
- *PRCID*
- *PRGID*
- *PRFB*
- *PRACC*

Les zones absentes n'occupent aucun espace dans l'enregistrement.

Étant donné que MQPMR n'a pas de présentation fixe, aucune définition de cette dernière n'est fournie dans le fichier COPY. Le programmeur d'application doit créer une déclaration contenant les zones requises par l'application et définir les indicateurs dans *PMPRF* pour indiquer les zones présentes.

- «Zones», à la page 1247
- «Valeurs initiales», à la page 1248
- «Déclaration RPG», à la page 1248

## Zones

La structure MQPMR contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

### PRACC (chaîne de 32 octets)

Jeton de comptabilité.

Il s'agit du jeton de comptabilité à utiliser pour le message envoyé à la file d'attente avec un nom qui a été spécifié par l'élément correspondant dans le tableau des structures MQOR fournies sur l'appel MQOPEN ou MQPUT1 . Il est traité de la même manière que la zone *MDACC* dans MQMD pour une insertion dans une file d'attente unique. Voir la description de *MDACC* dans «MQMD (descripteur de message) sous IBM i», à la page 1162 pour plus d'informations sur le contenu de cette zone.

Si cette zone n'est pas présente, la valeur de MQMD est utilisée.

Il s'agit d'une zone d'entrée.

### PRCID (chaîne de 24 octets)

Identificateur de corrélation.

Il s'agit de l'identificateur de corrélation à utiliser pour le message envoyé à la file d'attente avec le nom qui a été spécifié par l'élément correspondant dans le tableau des structures MQOR fournies sur l'appel MQOPEN ou MQPUT1 . Il est traité de la même manière que la zone *MDCID* dans MQMD pour une insertion dans une file d'attente unique.

Si cette zone n'est pas présente dans l'enregistrement MQPMR ou s'il y a moins d'enregistrements MQPMR que de destinations, la valeur de MQMD est utilisée pour les destinations qui n'ont pas d'enregistrement MQPMR contenant une zone *PRCID* .

Si PMNCID est spécifié, un nouvel identificateur de corrélation *unique* est généré et utilisé pour toutes les destinations de la liste de distribution, qu'elles comportent ou non des enregistrements MQPMR. Ceci est différent de la façon dont PMNMID est traité (voir la zone *PRMID* ).

Il s'agit d'une zone d'entrée/sortie.

### PRFB (entier signé à 10 chiffres)

Retour d'informations ou code anomalie.

Il s'agit du code retour à utiliser pour le message envoyé à la file d'attente avec le nom qui a été spécifié par l'élément correspondant dans le tableau des structures MQOR fournies sur l'appel MQOPEN ou MQPUT1 . Il est traité de la même manière que la zone *MDFB* dans MQMD pour une insertion dans une file d'attente unique.

Si cette zone n'est pas présente, la valeur de MQMD est utilisée.

Il s'agit d'une zone d'entrée.

### PRGID (chaîne de 24 octets)

Identificateur de groupe.

Il s'agit de l'identificateur de groupe à utiliser pour le message envoyé à la file d'attente avec le nom spécifié par l'élément correspondant dans le tableau de structures MQOR fourni sur l'appel MQOPEN ou MQPUT1 . Il est traité de la même manière que la zone *MDGID* dans MQMD pour une insertion dans une file d'attente unique.

Si cette zone n'est pas présente dans l'enregistrement MQPMR ou s'il y a moins d'enregistrements MQPMR que de destinations, la valeur de MQMD est utilisée pour les destinations qui n'ont pas d'enregistrement MQPMR contenant une zone *PRGID* . La valeur est traitée comme indiqué dans [Tableau 716, à la page 1234](#), mais avec les différences suivantes:

- Dans les cas où un nouvel identificateur de groupe serait utilisé, le gestionnaire de files d'attente génère un identificateur de groupe différent pour chaque destination (c'est-à-dire que deux destinations n'ont pas le même identificateur de groupe).
- Dans les cas où la valeur de la zone serait utilisée, l'appel échoue avec le code anomalie RC2258.

Il s'agit d'une zone d'entrée/sortie.

### PRMID (chaîne de 24 octets)

Identificateur de message.

Il s'agit de l'identificateur de message à utiliser pour le message envoyé à la file d'attente avec le nom spécifié par l'élément correspondant dans le tableau de structures MQOR fourni sur l'appel MQOPEN ou MQPUT1. Il est traité de la même manière que la zone *MDMID* dans MQMD pour une insertion dans une file d'attente unique.

Si cette zone n'est pas présente dans l'enregistrement MQPMR ou s'il y a moins d'enregistrements MQPMR que de destinations, la valeur de MQMD est utilisée pour les destinations qui n'ont pas d'enregistrement MQPMR contenant une zone *PRMID*. Si cette valeur est MINONE, un nouvel identificateur de message est généré pour *chaque* de ces destinations (c'est-à-dire que deux de ces destinations n'ont pas le même identificateur de message).

Si PMNMID est spécifié, de nouveaux identificateurs de message sont générés pour toutes les destinations de la liste de distribution, qu'elles comportent ou non des enregistrements MQPMR. Ceci est différent de la façon dont PMNCID est traité (voir la zone *PRCID*).

Il s'agit d'une zone d'entrée/sortie.

### Valeurs initiales

Aucune valeur initiale n'est définie pour cette structure, car aucune déclaration de structure n'est fournie. L'exemple de déclaration suivant montre comment la structure doit être déclarée par le programmeur d'application si toutes les zones sont requises.

### Déclaration RPG

```
D*..1.....2.....3.....4.....5.....6.....7..
D* MQPMR Structure
D*
D* Message identifier
D PRMID           1      24
D* Correlation identifier
D PRCID           25     48
D* Group identifier
D PRGID           49     72
D* Feedback or reason code
D PRFB            73     76I 0
D* Accounting token
D PRACC           77     108
```

## IBM i MQRFH (règles et en-tête de formatage) sous IBM i

La structure MQRFH définit la présentation des règles et l'en-tête de formatage.

### Présentation

**Objectif:** Cet en-tête peut être utilisé pour envoyer des données de chaîne sous la forme de paires nom-valeur.

**Nom de format:** FMRFH.

**Jeu de caractères et codage:** les zones de la structure MQRFH (y compris *RFNVS*) se trouvent dans le jeu de caractères et le codage donnés par les zones *MDCSI* et *MDENC* de la structure d'en-tête qui précède



MQRFH, ou par ces zones de la structure MQMD si MQRFH se trouve au début des données de message d'application.

Le jeu de caractères doit comporter des caractères mono-octet pour les caractères admis dans les noms de file d'attente.

- «Zones», à la page 1249
- «Valeurs initiales», à la page 1251
- «Déclaration RPG», à la page 1251

## Zones

La structure MQRFH contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

### **RFCSI (entier signé à 10 chiffres)**

Identificateur de jeu de caractères des données qui suivent *RFNVS*.

Indique l'identificateur de jeu de caractères des données qui suivent *RFNVS* ; elle ne s'applique pas aux données de type caractères dans la structure MQRFH elle-même.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données. La valeur spéciale suivante peut être utilisée:

#### **CSINHT**

Hérite de l'identificateur de jeu de caractères de cette structure.

Les données de type caractères dans les données *suivant* cette structure se trouvent dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, la valeur CSINHT n'est pas renvoyée par l'appel MQGET.

CSINHT ne peut pas être utilisé si la valeur de la zone *MDPAT* dans MQMD est ATBRKR.

La valeur initiale de cette zone est CSUNDF.

Codage numérique des données qui suit *RFNVS*.

Indique le codage numérique des données qui suivent *RFNVS* ; elle ne s'applique pas aux données numériques de la structure MQRFH elle-même.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données.

La valeur initiale de cette zone est ENNAT.

### **RFFLG (entier signé à 10 chiffres)**

Indicateurs.

Les éléments suivants peuvent être spécifiés:

#### **RFNONE**

Aucun indicateur.

La valeur initiale de cette zone est RFNONE.

### **RFFMT (chaîne de caractères de 8 octets)**

Nom de format des données qui suivent *RFNVS*.

Indique le nom de format des données qui suivent *RFNVS*.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de la zone *MDFMT* dans MQMD.

La valeur initiale de cette zone est FMNONE.

### **RFLEN (entier signé à 10 chiffres)**

Longueur totale de MQRFH incluant *RFNVS*.

Il s'agit de la longueur en octets de la structure MQRFH, y compris la zone *RFNVS* à la fin de la structure. La longueur n'inclut pas les données utilisateur qui suivent la zone *RFNVS*.

Pour éviter les problèmes liés à la conversion des données utilisateur dans certains environnements, envisagez d'utiliser *RFLEN* comme multiple de quatre.

La constante suivante indique la longueur de la partie *fixe* de la structure, c'est-à-dire la longueur à l'exclusion de la zone *RFNVS* :

#### **ENV RFLV**

Longueur de la partie fixe de la structure MQRFH.

La valeur initiale de cette zone est RFLENV.

### **RFNVS (chaîne de caractères à n octets)**

Chaîne contenant des paires nom-valeur.

Il s'agit d'une chaîne de caractères de longueur variable contenant des paires nom-valeur au format suivant:

```
name1 value1 name2 value2 name3 value3 ...
```

Chaque nom ou valeur doit être séparé du nom ou de la valeur adjacente par un ou plusieurs caractères blancs ; ces blancs ne sont pas significatifs. Un nom ou une valeur peut contenir des blancs significatifs en ajoutant un préfixe et un suffixe au nom ou à la valeur avec le caractère de guillemet ; tous les caractères entre le guillemet ouvrant et le guillemet fermant correspondant sont considérés comme significatifs. Dans l'exemple suivant, le nom est FAMOUS\_WORDS et la valeur est Hello World:

```
FAMOUS_WORDS "Hello World"
```

Un nom ou une valeur peut contenir des caractères autres que le caractère null (qui sert de délimiteur pour *RFNVS*). Toutefois, pour faciliter l'interopérabilité, une application peut préférer limiter les noms aux caractères suivants:

- Premier caractère: lettre majuscule ou minuscule (A à Z ou a à z) ou trait de soulignement.
- Caractères suivants: caractères alphabétiques majuscules ou minuscules, chiffres décimaux (0 à 9), trait de soulignement, trait d'union ou point.

Si un nom ou une valeur contient un ou plusieurs guillemets, le nom ou la valeur doit être placé entre guillemets et chaque guillemet dans la chaîne doit être doublé:

```
Famous_Words "The program displayed ""Hello World"""
```

Les noms et les valeurs sont sensibles à la casse, c'est-à-dire que les minuscules ne sont pas considérées comme des majuscules. Par exemple, FAMOUS\_WORDS et Famous\_Words sont deux noms différents.

La longueur en octets de *RFNVS* est égale à *RFLEN* moins *RFLENV*. Pour éviter les problèmes liés à la conversion des données utilisateur dans certains environnements, il est recommandé que cette longueur soit un multiple de quatre. *RFNVS* doit être rempli avec des blancs de cette longueur ou se terminer plus tôt en plaçant un caractère null après le dernier caractère significatif de la chaîne. Le caractère null et les octets qui le suivent, jusqu'à la longueur spécifiée de *RFNVS*, sont ignorés.

**Remarque :** Comme la longueur de cette zone n'est pas fixe, elle est omise dans les déclarations de la structure fournies pour les langages de programmation pris en charge.

### RFSID (chaîne de caractères de 4 octets)

Identificateur de structure.

La valeur doit être:

#### RFSIDV

Identificateur des règles et de la structure d'en-tête de formatage.

La valeur initiale de ce champ est RFSIDV.

### RFVER (entier signé à 10 chiffres)

Numéro de version de la structure.

La valeur doit être:

#### RFVER1

Règles Version-1 et structure d'en-tête de formatage.

La valeur initiale de ce champ est RFVER1.

## Valeurs initiales

Tableau 720. Champs dans MQRFH		
Nom de zone	Nom de la constante	Valeur de la constante
RFSID	RFSIDV	'RFH↵'
RFVER	RFVER1	1
RFLEN	ENV RFLV	32
RFENC	ENNAT	Dépend de l'environnement
RFCSI	CSUNDF	0
RFFMT	FMAUCUN	Espaces vides
RFFLG	RFNONE	0

**Remarques :**

1. Le symbole ↵ représente un caractère blanc unique.

## Déclaration RPG

```
D*.1.....2.....3.....4.....5.....6.....7..
D* MQRFH Structure
D*
D* Structure identifier
D RFSID          1      4    INZ('RFH ')
D* Structure version number
D RFVER          5      8I 0 INZ(1)
D* Total length of MQRFH includingNameValueString
D RFLEN          9     12I 0 INZ(32)
D* Numeric encoding of data that followsNameValueString
D RFENC         13     16I 0 INZ(273)
D* Character set identifier of data thatfollows NameValueString
D RFCSI         17     20I 0 INZ(0)
D* Format name of data that followsNameValueString
D RFFMT         21     28    INZ(' ')
D* Flags
D RFFLG         29     32I 0 INZ(0)
```

La structure MQRFH2 définit le format des règles version-2 et l'en-tête de formatage.

## Présentation

**Objectif:** Cet en-tête peut être utilisé pour envoyer des données qui ont été codées à l'aide d'une syntaxe de type XML. Un message peut contenir au moins deux structures MQRFH2 en série, les données utilisateur pouvant éventuellement suivre la dernière structure MQRFH2 de la série.

**Nom de format:** FMRFH2.

**Jeu de caractères et codage:** des règles spéciales s'appliquent au jeu de caractères et au codage utilisés pour la structure MQRFH2 :

- Les zones autres que *RF2NVD* se trouvent dans le jeu de caractères et le codage donnés par les zones *MDCSI* et *MDENC* de la structure d'en-tête qui précède MQRFH2, ou par ces zones de la structure MQMD si MQRFH2 se trouve au début des données du message d'application.

Le jeu de caractères doit comporter des caractères mono-octet pour les caractères admis dans les noms de file d'attente.

Lorsque GMCONV est spécifié dans l'appel MQGET, le gestionnaire de files d'attente convertit ces zones en jeu de caractères et codage requis.

- *RF2NVD* correspond au jeu de caractères indiqué par la zone *RF2NVC* . Seuls certains jeux de caractères Unicode sont valides pour *RF2NVC* (voir la description de *RF2NVC* pour plus de détails).

Certains jeux de caractères ont une représentation qui dépend du codage. Si *RF2NVC* est l'un de ces jeux de caractères, *RF2NVD* doit être dans le même codage que les autres zones de MQRFH2.

Lorsque GMCONV est spécifié dans l'appel MQGET, le gestionnaire de files d'attente convertit *RF2NVD* en codage demandé, mais ne modifie pas son jeu de caractères.

- «Zones», à la page [1252](#)
- «Valeurs initiales», à la page [1257](#)
- «Déclaration RPG», à la page [1258](#)

## Zones

La structure MQRFH2 contient les zones suivantes ; elles sont décrites par ordre alphabétique:

### **RF2CSI (entier signé à 10 chiffres)**

Identificateur de jeu de caractères des données qui suivent la dernière zone *RF2NVD* .

Indique l'identificateur de jeu de caractères des données qui suivent la dernière zone *RF2NVD* . Elle ne s'applique pas aux données de type caractères de la structure MQRFH2 elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. La valeur spéciale suivante peut être utilisée:

#### **CSINHT**

Hérite de l'identificateur de jeu de caractères de cette structure.

Les données de type caractères dans les données *suivant* cette structure se trouvent dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, la valeur CSINHT n'est pas renvoyée par l'appel MQGET.

CSINHT ne peut pas être utilisé si la valeur de la zone *MDPAT* dans MQMD est ATBRKR.

La valeur initiale de cette zone est CSINHT.

### **RF2ENC (entier signé à 10 chiffres)**

Codage numérique des données qui suit la dernière zone *RF2NVD* .

Indique le codage numérique des données qui suivent la dernière zone *RF2NVD* ; il ne s'applique pas aux données numériques de la structure *MQRFH2* elle-même.

Dans l'appel *MQPUT* ou *MQPUT1* , l'application doit définir cette zone sur la valeur appropriée aux données.

La valeur initiale de cette zone est *ENNAT*.

### **RF2FLG (entier signé à 10 chiffres)**

Indicateurs.

La valeur suivante doit être spécifiée:

#### **RFNONE**

Aucun indicateur.

La valeur initiale de cette zone est *RFNONE*.

### **RF2FMT (chaîne de caractères de 8 octets)**

Nom de format des données qui suivent la dernière zone *RF2NVD* .

Indique le nom de format des données qui suivent la dernière zone *RF2NVD* .

Dans l'appel *MQPUT* ou *MQPUT1* , l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de la zone *MDFMT* dans *MQMD*.

La valeur initiale de cette zone est *FMNONE*.

### **RF2LEN (entier signé à 10 chiffres)**

Longueur totale de *MQRFH2* incluant toutes les zones *RF2NVL* et *RF2NVD* .

Il s'agit de la longueur en octets de la structure *MQRFH2* , y compris les zones *RF2NVL* et *RF2NVD* à la fin de la structure. Il est possible qu'il y ait plusieurs paires de zones *RF2NVL* et *RF2NVD* à la fin de la structure, dans la séquence:

```
length1, data1, length2, data2, ...
```

*RF2LEN* n'inclut pas les données utilisateur qui peuvent suivre la dernière zone *RF2NVD* à la fin de la structure.

Pour éviter les problèmes liés à la conversion des données utilisateur dans certains environnements, envisagez d'utiliser *RF2LEN* comme multiple de quatre.

La constante suivante indique la longueur de la partie *fixe* de la structure, c'est-à-dire la longueur à l'exclusion des zones *RF2NVL* et *RF2NVD* :

#### **RFLEN2**

Longueur de la partie fixe de la structure *MQRFH2* .

La valeur initiale de cette zone est *RFLEN2*.

### **RF2NVC (entier signé à 10 chiffres)**

Identificateur de jeu de caractères de *RF2NVD*.

Indique l'identificateur de jeu de caractères codés des données de la zone *RF2NVD* . Il est différent du jeu de caractères des autres chaînes de la structure *MQRFH2* et peut être différent du jeu de caractères des données (le cas échéant) qui suit la dernière zone *RF2NVD* à la fin de la structure.

*RF2NVC* doit avoir l'une des valeurs de *CCSID* suivantes:

**1 200**

UTF-16, version Unicode la plus récente prise en charge

**13488**

UTF-16, sous-ensemble de la version Unicode 2.0

**17584**

UTF-16, Unicode version 3.0 sous-ensemble (inclut le symbole Euro)

**1208**

UTF-8, version Unicode la plus récente prise en charge

Pour les jeux de caractères UTF-16 , le codage (ordre des octets) de *RF2NVD* doit être identique au codage des autres zones de la structure MQRFH2 . Les caractères de substitution (X'D800' à X'DFFF') ne sont pas pris en charge.

**Remarque :** Si *RF2NVC* ne possède pas l'une des valeurs répertoriées précédemment et que la structure MQRFH2 requiert une conversion sur l'appel MQGET, l'appel se termine avec le code anomalie RC2111 et le message est renvoyé non converti.

La valeur initiale de cette zone est 1208.

**RF2NVD (chaîne de caractères de n octets)**

Données de nom / valeur.

Il s'agit d'une chaîne de caractères de longueur variable contenant des données codées à l'aide d'une syntaxe de type XML. La longueur en octets de cette chaîne est fournie par la zone *RF2NVL* qui précède la zone *RF2NVD* ; cette longueur doit être un multiple de quatre.

Les zones *RF2NVL* et *RF2NVD* sont facultatives, mais si elles sont présentes, elles doivent apparaître sous la forme d'une paire et être adjacentes. La paire de champs peut être répétée autant de fois que nécessaire, par exemple:

```
length1 data1 length2 data2 length3 data3
```

Etant donné que ces zones sont facultatives, elles sont omises dans les déclarations de la structure fournies pour les différents langages de programmation pris en charge.

*RF2NVD* est inhabituel car il n'est pas converti au jeu de caractères spécifié dans l'appel MQGET lorsque le message est extrait avec l'option GMCONV activée ; *RF2NVD* reste dans son jeu de caractères d'origine. Toutefois, *RF2NVD* est converti au codage spécifié dans l'appel MQGET.

**Syntaxe des données de nom / valeur:** la chaîne se compose d'un "dossier" unique qui contient zéro ou plusieurs propriétés. Le dossier est délimité par des balises de début et de fin XML portant le même nom que le dossier:

```
<folder> property1 property2 ... </folder>
```

Les caractères qui suivent la balise de fin de dossier, jusqu'à la longueur définie par *RF2NVL*, doivent être vides. Dans le dossier, chaque propriété est composée d'un nom et d'une valeur, et éventuellement d'un type de données:

```
<name dt="datatype">value</name>
```

Dans ces exemples:

- Les caractères délimiteurs (<, =, ", / et >) doivent être spécifiés exactement comme indiqué.
- name est le nom spécifié par l'utilisateur de la propriété ; voir l'exemple suivant pour plus d'informations sur les noms.
- datatype est un type de données facultatif spécifié par l'utilisateur de la propriété ; voir l'exemple suivant pour les types de données valides.

- value est la valeur définie par l'utilisateur de la propriété. Pour plus d'informations sur les valeurs, voir les paragraphes suivants.
- Les blancs sont significatifs entre le caractère > qui précède une valeur et le caractère < qui suit la valeur, et au moins un blanc doit précéder dt=. Ailleurs, les blancs peuvent être codés librement entre les étiquettes, ou entre les étiquettes précédentes ou suivantes (par exemple pour améliorer la lisibilité) ; ces blancs ne sont pas significatifs.

Si les propriétés sont liées les unes aux autres, elles peuvent être regroupées en les plaçant dans des balises XML de début et de fin portant le même nom que le groupe:

```
<folder> <group> property1 property2 ... </group> </folder>
```

Les groupes peuvent être imbriqués dans d'autres groupes, sans limite, et un groupe peut apparaître plusieurs fois dans un dossier. Il est également possible qu'un dossier contienne des propriétés dans des groupes et d'autres propriétés ne figurant pas dans des groupes.

**Noms des propriétés, des groupes et des dossiers:** les noms des propriétés, des groupes et des dossiers doivent être des noms de balises XML valides, à l'exception du caractère deux-points, qui n'est pas autorisé dans un nom de propriété, de groupe ou de dossier. Notamment :

- Les noms doivent commencer par une lettre ou un trait de soulignement. Les lettres valides sont définies dans la spécification XML W3C et se composent essentiellement des catégories Unicode Ll, Lu, Lo, Lt et Nl.
- Les caractères restants d'un nom peuvent être des lettres, des décimales, des traits de soulignement, des traits d'union ou des points. Elles correspondent aux catégories Unicode Ll, Lu, Lo, Lt, Nl, Mc, Mn, Lm et Nd.
- Les caractères de compatibilité Unicode (X'F900'et versions ultérieures) ne sont autorisés dans aucune partie d'un nom.
- Les noms ne doivent pas commencer par la chaîne XML en minuscules ou en majuscules.

De plus :

- Il est nécessaire de respecter l'utilisation des minuscules et des majuscules dans les noms. Par exemple, ABC, abcet Abc sont trois noms différents.
- Chaque dossier possède un espace de nom distinct. Par conséquent, un groupe ou une propriété d'un dossier n'est pas en conflit avec un groupe ou une propriété du même nom dans un autre dossier.
- Les groupes et les propriétés occupent le même espace de nom dans un dossier. Par conséquent, une propriété ne peut pas avoir le même nom qu'un groupe dans le dossier contenant cette propriété.

En règle générale, les programmes qui analysent la zone *RF2NVD* doivent ignorer les propriétés ou les groupes dont les noms ne sont pas reconnus par le programme, à condition que ces propriétés ou ces groupes soient correctement formés.

**Types de données des propriétés:** chaque propriété peut avoir un type de données facultatif. S'il est spécifié, le type de données doit être l'une des valeurs suivantes, en majuscules, en minuscules ou en casse mixte:

Type de données	Utilisé pour
string	N'importe quelle séquence de caractères. Certains caractères doivent être spécifiés à l'aide de séquences d'échappement.
boolean	Le caractère 0 ou 1 (1 indique TRUE).
bin.hex	Chiffres hexadécimaux représentant les octets.

<i>Tableau 721. Types de données et leur utilisation (suite)</i>	
Type de données	Utilisé pour
i1	Nombre entier compris entre -128 et +127, exprimé uniquement à l'aide de chiffres décimaux et d'un signe facultatif.
i2	Nombre entier compris entre -32 768 et +32 767, exprimé uniquement à l'aide de chiffres décimaux et d'un signe facultatif.
i4	Nombre entier compris entre -2 147 483 648 et + 2 147 483 647, exprimé uniquement à l'aide de chiffres décimaux et d'un signe facultatif.
i8	Nombre entier compris entre -9 223 372 036 854 775 808 et + 9 223 372 036 854 775 807, exprimé en utilisant uniquement des chiffres décimaux et un signe facultatif.
int	Nombre entier compris entre -9 223 372 036 854 775 808 et + 9 223 372 036 854 775 807, exprimé en utilisant uniquement des chiffres décimaux et un signe facultatif. Il peut être utilisé à la place de i1, i2, i4 ou i8 si l'expéditeur ne souhaite pas impliquer une précision particulière.
r4	Nombre à virgule flottante dont la magnitude est comprise entre 1.175E-37 et 3.402 823 47E+38, exprimé à l'aide de chiffres décimaux, de signes facultatifs, de chiffres fractionnaires facultatifs et d'exposant facultatif.
r8	Nombre à virgule flottante dont la magnitude est comprise entre 2.225E-307 et 1.797 693 134 862 3E+308, exprimé à l'aide de chiffres décimaux, de signes facultatifs, de chiffres fractionnaires facultatifs et d'un exposant facultatif.

**Valeurs des propriétés:** la valeur d'une propriété peut être constituée de n'importe quels caractères, à l'exception des caractères spéciaux auxquels une séquence d'échappement obligatoire est associée. Chaque occurrence de la valeur d'un caractère marqué comme "obligatoire" dans le tableau suivant doit être remplacée par la séquence d'échappement correspondante. Le tableau affiche également les caractères associés à une séquence d'échappement facultative. Chaque occurrence dans la valeur d'un caractère marqué comme "facultatif" peut être remplacée par la séquence d'échappement correspondante, mais cela n'est pas obligatoire.

<i>Tableau 722. Caractères d'échappement et leur utilisation</i>		
Caractère	Séquence d'échappement	Utilisation
&	&amp;	Obligatoire
<	<	Obligatoire
>	&gt;	Facultatif
"	&quot;	Facultatif
'	&apos;	Facultatif



**Remarque :** Le caractère & au début d'une séquence d'échappement ne doit pas être remplacé par &amp; ; .

Dans l'exemple suivant, les blancs de la valeur sont significatifs ; cependant, aucune séquence d'échappement n'est nécessaire:

```
<Famous_Words>The program displayed "Hello World"</Famous_Words>
```

### RF2NVL (entier signé à 10 chiffres)

Longueur de *RF2NVD*.

Indique la longueur en octets des données de la zone *RF2NVD* . Pour éviter les problèmes liés à la conversion des données (le cas échéant) qui suivent la zone *RF2NVD* , *RF2NVL* doit être un multiple de quatre.

**Remarque :** Les zones *RF2NVL* et *RF2NVD* sont facultatives, mais si elles sont présentes, elles doivent apparaître sous la forme d'une paire et être adjacentes. La paire de champs peut être répétée autant de fois que nécessaire, par exemple:

```
length1 data1 length2 data2 length3 data3
```

Etant donné que ces zones sont facultatives, elles sont omises dans les déclarations de la structure fournies pour les différents langages de programmation pris en charge.

### RF2SID (chaîne de caractères de 4 octets)

Identificateur de structure.

La valeur doit être:

#### RFSIDV

Identificateur des règles et de la structure d'en-tête de formatage.

La valeur initiale de ce champ est RFSIDV.

### RF2VER (entier signé à 10 chiffres)

Numéro de version de la structure.

La valeur doit être:

#### RFVER2

Règles Version-2 et structure d'en-tête de formatage.

La valeur initiale de ce champ est RFVER2.

## Valeurs initiales

Nom de zone	Nom de la constante	Valeur de la constante
<i>RF2SID</i>	RFSIDV	'RFH→'
<i>RF2VER</i>	RFVER2	2
<i>RF2LEN</i>	RFLLEN2	36
<i>RF2ENC</i>	ENNAT	Dépend de l'environnement
<i>RF2CSI</i>	CSINHT	-2
<i>RF2FMT</i>	FMAUCUN	Espaces vides
<i>RF2FLG</i>	RFNONE	0

Tableau 723. Zones dans MQRFH2 (suite)

Nom de zone	Nom de la constante	Valeur de la constante
RF2NVC	Aucun	1208

**Remarques :**

1. Le symbole – représente un caractère blanc unique.

**Déclaration RPG**

```

D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQRFH2 Structure
D*
D* Structure identifier
D RF2SID          1          4    INZ('RFH ')
D* Structure version number
D RF2VER          5          8I 0 INZ(2)
D* Total length of MQRFH2 including allNameValueLength and
D* NameValueDatafields
D RF2LEN          9          12I 0 INZ(36)
D* Numeric encoding of data that followslast NameValueData field
D RF2ENC          13         16I 0 INZ(273)
D* Character set identifier of data thatfollows last NameValueData field
D RF2CSI          17         20I 0 INZ(-2)
D* Format name of data that follows lastNameValueData field
D RF2FMT          21         28    INZ(' ')
D* Flags
D RF2FLG          29         32I 0 INZ(0)
D* Character set identifier ofNameValueData
D RF2NVC          33         36I 0 INZ(1208)

```

**MQRMH (en-tête de message de référence) sur IBM i**

La structure MQRMH définit le format d'un en-tête de message de référence.

**Présentation**

**Objectif:** Cet en-tête est utilisé avec les exits de canal de message écrits par l'utilisateur pour envoyer de grandes quantités de données (appelées "données non formatées") d'un gestionnaire de files d'attente à un autre. La différence par rapport à la messagerie normale est que les données non formatées ne sont pas stockées dans une file d'attente ; à la place, seule une *référence* aux données non formatées est stockée dans la file d'attente. Cela réduit la possibilité que les ressources IBM MQ soient épuisées par quelques messages de grande taille.

**Nom de format:** FMRMH.

**Jeu de caractères et codage:** les données de type caractère dans MQRMH et les chaînes traitées par les zones de décalage doivent figurer dans le jeu de caractères du gestionnaire de files d'attente local ; cet attribut est indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** . Les données numériques dans MQRMH doivent être dans le codage de la machine native ; elles sont fournies par la valeur de ENNAT pour le langage de programmation C.

Le jeu de caractères et le codage du MQRMH doivent être définis dans les zones *MDCSI* et *MDENC* dans:

- MQMD (si la structure MQRMH est au début des données de message), ou
- Structure d'en-tête qui précède la structure MQRMH (tous les autres cas).

**Utilisation:** une application insère un message composé d'un MQRMH, mais en omettant les données non formatées. Lorsque le message est lu à partir de la file d'attente de transmission par un agent MCA, un exit de message fourni par l'utilisateur est appelé pour traiter l'en-tête de message de référence. L'exit peut ajouter au message de référence les données non formatées identifiées par la structure MQRMH, avant que l'agent MCA n'envoie le message via le canal au gestionnaire de files d'attente suivant.

A la fin de la réception, un exit de message qui attend les messages de référence doit exister. Lorsqu'un message de référence est reçu, l'exit doit créer l'objet à partir des données non formatées qui suivent le MQRMH dans le message, puis transmettre le message de référence sans les données non formatées. Le message de référence peut ensuite être extrait par une application lisant le message de référence (sans les données non formatées) à partir d'une file d'attente.

Normalement, la structure MQRMH est tout ce qui se trouve dans le message. Toutefois, si le message se trouve dans une file d'attente de transmission, un ou plusieurs en-têtes supplémentaires précèdent la structure MQRMH.

Un message de référence peut également être envoyé à une liste de distribution. Dans ce cas, la structure MQDH et ses enregistrements associés précèdent la structure MQRMH lorsque le message se trouve dans une file d'attente de transmission.

**Remarque :** Un message de référence ne doit pas être envoyé en tant que message segmenté, car l'exit de message ne peut pas le traiter correctement.

- «Conversion de données», à la page [1259](#)
- «Zones», à la page [1259](#)
- «Valeurs initiales», à la page [1263](#)
- «Déclaration RPG», à la page [1265](#)

## Conversion de données

A des fins de conversion de données, la conversion de la structure MQRMH inclut la conversion des données d'environnement source, du nom d'objet source, des données d'environnement de destination et du nom d'objet de destination. Tous les autres octets compris dans *RMLLEN* octets du début de la structure sont supprimés ou ont des valeurs non définies après la conversion des données. Les données non formatées seront converties à condition que toutes les instructions suivantes soient vraies:

- Les données non formatées sont présentes dans le message lorsque la conversion de données est effectuée.
- La zone *RMFMT* de MQRMH a une valeur autre que FMNONE.
- Un exit de conversion de données écrit par l'utilisateur existe avec le nom de format indiqué.

Sachez toutefois que les données non formatées ne sont généralement pas présentes dans le message lorsque le message se trouve dans une file d'attente et que, par conséquent, les données non formatées ne seront pas converties par l'option GMCONV.

## Zones

La structure MQRMH contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

### RMCSI (entier signé à 10 chiffres)

Identificateur de jeu de caractères des données non formatées.

Indique l'identificateur de jeu de caractères des données non formatées ; il ne s'applique pas aux données de type caractères de la structure MQRMH elle-même.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données. La valeur spéciale suivante peut être utilisée:

#### CSINHT

Hérite de l'identificateur de jeu de caractères de cette structure.

Les données de type caractères dans les données *suivant* cette structure se trouvent dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, la valeur CSINHT n'est pas renvoyée par l'appel MQGET.

CSINHT ne peut pas être utilisé si la valeur de la zone *MDPAT* dans *MQMD* est *ATBRKR*.

La valeur initiale de cette zone est *CSUNDF*.

#### **RMDEL (entier signé à 10 chiffres)**

Longueur des données d'environnement de destination.

Si cette zone a pour valeur zéro, il n'y a pas de données d'environnement de destination et *RMDEO* est ignoré.

#### **RMDEO (entier signé à 10 chiffres)**

Décalage des données d'environnement de destination.

Cette zone indique le décalage des données d'environnement de destination à partir du début de la structure *MQRMH*. Les données d'environnement de destination peuvent être spécifiées par le créateur du message de référence, si ces données sont connues du créateur. Par exemple, les données d'environnement de destination peuvent être le chemin de répertoire de l'objet dans lequel les données non formatées doivent être stockées. Toutefois, si le créateur ne connaît pas les données d'environnement de destination, il incombe à l'exit de message fourni par l'utilisateur de déterminer les informations d'environnement requises.

La longueur des données de l'environnement de destination est indiquée par *RMDEL* ; Si cette longueur est égale à zéro, il n'y a pas de données d'environnement de destination et *RMDEO* est ignoré. S'il est présent, les données de l'environnement de destination doivent résider entièrement dans *RMLen* octets à partir du début de la structure.

Les applications ne doivent pas supposer que les données de l'environnement de destination sont contiguës à l'une des données traitées par les zones *RMSEO*, *RMSNO* et *RMDNO*.

La valeur initiale de cette zone est 0.

#### **RMDL (entier signé à 10 chiffres)**

Longueur des données non formatées.

La zone *RMDL* indique la longueur des données non formatées référencées par la structure *MQRMH*.

Si les données non formatées sont présentes dans le message, les données commencent à un décalage de *RMLen* octets à partir du début de la structure *MQRMH*. La longueur de l'ensemble du message moins *RMLen* indique la longueur des données non formatées présentes.

Si des données sont présentes dans le message, *RMDL* indique la quantité de données pertinentes. Dans le cas normal, *RMDL* doit avoir la même valeur que la longueur des données présentes dans le message.

Si la structure *MQRMH* représente les données restantes dans l'objet (à partir du décalage logique spécifié), la valeur zéro peut être utilisée pour *RMDL*, si les données non formatées ne sont pas présentes dans le message.

Si aucune donnée n'est présente, la fin de *MQRMH* coïncide avec la fin du message.

La valeur initiale de cette zone est 0.

#### **RMDNL (entier signé à 10 chiffres)**

Longueur du nom de l'objet de destination.

Si cette zone est à zéro, il n'y a pas de nom d'objet de destination et *RMDNO* est ignoré.

#### **RMDNO (entier signé à 10 chiffres)**

Décalage du nom de l'objet de destination.

Cette zone indique le décalage du nom d'objet de destination à partir du début de la structure *MQRMH*. Le nom de l'objet de destination peut être spécifié par le créateur du message de référence, si ces données sont connues du créateur. Toutefois, si le créateur ne connaît pas le nom de l'objet de destination, il incombe à l'exit de message fourni par l'utilisateur d'identifier l'objet à créer ou à modifier.

La longueur du nom de l'objet de destination est donnée par *RMDNL* ; Si cette longueur est égale à zéro, il n'y a pas de nom d'objet de destination et *RMDNO* est ignoré. S'il est présent, le nom de l'objet de destination doit résider entièrement dans *RMLLEN* octets à partir du début de la structure.

Les applications ne doivent pas supposer que le nom de l'objet de destination est contigu à l'une des données traitées par les zones *RMSEO*, *RMSNO* et *RMDEO*.

La valeur initiale de cette zone est 0.

### **RMDO (entier signé à 10 chiffres)**

Décalage faible des données non formatées.

Cette zone indique le décalage faible des données non formatées à partir du début de l'objet dont font partie les données non formatées. Le décalage des données non formatées à partir du début de l'objet est appelé *décalage logique*. Il ne s'agit pas du décalage physique des données non formatées à partir du début de la structure MQRMH-ce décalage est fourni par *RMLLEN*.

Pour permettre l'envoi d'objets volumineux à l'aide de messages de référence, le décalage logique est divisé en deux zones et le décalage logique réel est donné par la somme de ces deux zones:

- *RMDO* représente le reste obtenu lorsque le décalage logique est divisé par 1 000 000 000. Il s'agit donc d'une valeur comprise entre 0 et 999 999 999.
- *RMDO2* représente le résultat obtenu lorsque le décalage logique est divisé par 1 000 000 000. C'est donc le nombre de multiples complets de 1 000 000 000 qui existent dans le décalage logique. Le nombre de multiples est compris entre 0 et 999 999 999.

La valeur initiale de cette zone est 0.

### **RMDO2 (entier signé à 10 chiffres)**

Décalage élevé des données non formatées.

Cette zone indique le décalage élevé des données non formatées à partir du début de l'objet dont font partie les données non formatées. Il s'agit d'une valeur comprise entre 0 et 999 999 999. Voir *RMDO* pour des détails.

La valeur initiale de cette zone est 0.

### **RMENC (entier signé à 10 chiffres)**

Codage numérique des données non formatées.

Indique le codage numérique des données non formatées ; il ne s'applique pas aux données numériques de la structure MQRMH elle-même.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données.

La valeur initiale de cette zone est ENNAT.

### **RMFLG (entier signé à 10 chiffres)**

Indicateurs de message de référence.

Les indicateurs suivants sont définis:

#### **RMLAST**

Le message de référence contient ou représente la dernière partie de l'objet.

Cet indicateur indique que le message de référence représente ou contient la dernière partie de l'objet référencé.

#### **RRMNLST**

Le message de référence ne contient pas ou ne représente pas la dernière partie de l'objet.

RRMNLST est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette option soit utilisée avec une autre option, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

La valeur initiale de cette zone est RMNLST.

### **RMFMT (chaîne de caractères de 8 octets)**

Nom de format des données non formatées.

Indique le nom de format des données non formatées.

Dans l'appel MQPUT ou MQPUT1, l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de la zone *MDFMT* dans MQMD.

La valeur initiale de cette zone est FMNONE.

### **RMLLEN (entier signé à 10 chiffres)**

Longueur totale de MQRMH, y compris les chaînes à la fin des zones fixes, mais pas les données non formatées.

La valeur initiale de cette zone est zéro.

### **RMOII (chaîne de 24 octets)**

Identificateur de l'instance d'objet.

Cette zone peut être utilisée pour identifier une instance spécifique d'un objet. S'il n'est pas nécessaire, il doit être défini sur la valeur suivante:

#### **OIINON**

Aucun identificateur d'instance d'objet spécifié.

La valeur est zéro binaire pour la longueur de la zone.

La longueur de cette zone est indiquée par LNOIID. La valeur initiale de cette zone est OIINON.

### **RMOT (chaîne de caractères de 8 octets)**

Type d'objet.

Il s'agit d'un nom qui peut être utilisé par l'exit de message pour reconnaître les types de message de référence qu'il prend en charge. Pensez à rendre le nom conforme aux mêmes règles que la zone *RMFMT*.

La valeur initiale de cette zone est 8 blancs.

### **RMSEL (entier signé à 10 chiffres)**

Longueur des données d'environnement source.

Si cette zone est définie sur zéro, il n'y a pas de données d'environnement source et *RMSEO* est ignoré.

La valeur initiale de cette zone est 0.

### **RMSEO (entier signé à 10 chiffres)**

Décalage des données d'environnement source.

Cette zone indique le décalage des données d'environnement source à partir du début de la structure MQRMH. Les données d'environnement source peuvent être spécifiées par le créateur du message de référence, si ces données sont connues du créateur. Par exemple, les données d'environnement source peuvent être le chemin de répertoire de l'objet contenant les données non formatées. Toutefois, si le créateur ne connaît pas les données d'environnement source, il incombe à l'exit de message fourni par l'utilisateur de déterminer les informations d'environnement requises.

La longueur des données d'environnement source est indiquée par *RMSEL*; Si cette longueur est égale à zéro, il n'y a pas de données d'environnement source et *RMSEO* est ignoré. Si elles sont présentes, les données de l'environnement source doivent résider entièrement dans *RMLLEN* octets à partir du début de la structure.

Les applications ne doivent pas supposer que les données d'environnement démarrent immédiatement après la dernière zone fixe de la structure ou qu'elles sont contiguës à l'une des données traitées par les zones *RMSNO*, *RMDEO* et *RMDNO*.

La valeur initiale de cette zone est 0.

#### **RMSID (chaîne de caractères de 4 octets)**

Identificateur de structure.

La valeur doit être:

##### **RMSIDV**

Identificateur de la structure d'en-tête de message de référence.

La valeur initiale de cette zone est RMSIDV.

#### **RMSNL (entier signé à 10 chiffres)**

Longueur du nom de l'objet source.

Si cette zone est à zéro, il n'y a pas de nom d'objet source et *RMSNO* est ignoré.

La valeur initiale de cette zone est 0.

#### **RMSNO (entier signé à 10 chiffres)**

Décalage du nom de l'objet source.

Cette zone indique le décalage du nom d'objet source à partir du début de la structure *MQRMH*. Le nom de l'objet source peut être spécifié par le créateur du message de référence, si ces données sont connues du créateur. Toutefois, si le créateur ne connaît pas le nom de l'objet source, il incombe à l'exit de message fourni par l'utilisateur d'identifier l'objet auquel il doit accéder.

La longueur du nom d'objet source est donnée par *RMSNL* ; Si cette longueur est égale à zéro, il n'y a pas de nom d'objet source et *RMSNO* est ignoré. S'il est présent, le nom de l'objet source doit se trouver complètement dans *RMLLEN* octets à partir du début de la structure.

Les applications ne doivent pas supposer que le nom de l'objet source est contigu à l'une des données traitées par les zones *RMSEO*, *RMDEO* et *RMDNO*.

La valeur initiale de cette zone est 0.

#### **RMVER (entier signé à 10 chiffres)**

Numéro de version de la structure.

La valeur doit être:

##### **RMVER1**

La structure d'en-tête de message de référence Version-1.

La constante suivante indique le numéro de version de la version en cours:

##### **RMVERC**

Version actuelle de la structure d'en-tête de message de référence.

La valeur initiale de cette zone est RMVER1.

### **Valeurs initiales**

Nom de zone	Nom de la constante	Valeur de la constante
<i>RMSID</i>	RMSIDV	'RMH↵'
<i>RMVER</i>	RMVER1	1
<i>RMLLEN</i>	Aucun	0

Tableau 724. Zones dans MQRMH (suite)

Nom de zone	Nom de la constante	Valeur de la constante
RMENC	ENNAT	Dépend de l'environnement
RMCSI	CSUNDF	0
RMFMT	FMAUCUN	Espaces vides
RMFLG	RRMNLST	0
RMOT	Aucun	Espaces vides
RMOII	OIINON	Valeurs NULL
RMSEL	Aucun	0
RMSEO	Aucun	0
RMSNL	Aucun	0
RMSNO	Aucun	0
RMDEL	Aucun	0
RMDEO	Aucun	0
RMDNL	Aucun	0
RMDNO	Aucun	0
RMDL	Aucun	0
RMDO	Aucun	0
RMDO2	Aucun	0
<b>Remarques :</b>		
1. Le symbole ~ représente un caractère blanc unique.		

```

D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQRMH Structure
D*
D* Structure identifier
D RMSID          1      4    INZ('RMH ')
D* Structure version number
D RMVER          5      8I 0 INZ(1)
D* Total length of MQRMH, including strings at end of fixed fields, but not
D* the bulk data
D RMLEN          9     12I 0 INZ(0)
D* Numeric encoding of bulk data
D RMENC          13     16I 0 INZ(273)
D* Character set identifier of bulkdata
D RMCSI          17     20I 0 INZ(0)
D* Format name of bulk data
D RMFMT          21     28    INZ('      ')
D* Reference message flags
D RMFLG          29     32I 0 INZ(0)
D* Object type
D RMOT          33     40    INZ
D* Object instance identifier
D RMOII          41     64    INZ('00000000000000-
D                               00000000000000000000-
D                               000000000000')
D* Length of source environmentdata
D RMSEL          65     68I 0 INZ(0)
D* Offset of source environmentdata
D RMSEO          69     72I 0 INZ(0)
D* Length of source object name

```



```

D RMSNL          73      76I 0 INZ(0)
D* Offset of source object name
D RMSNO          77      80I 0 INZ(0)
D* Length of destination environmentdata
D RMDL           81      84I 0 INZ(0)
D* Offset of destination environmentdata
D RMDEO          85      88I 0 INZ(0)
D* Length of destination objectname
D RMDNL          89      92I 0 INZ(0)
D* Offset of destination objectname
D RMDNO          93      96I 0 INZ(0)
D* Length of bulk data
D RMDL           97     100I 0 INZ(0)
D* Low offset of bulk data
D RMDO          101     104I 0 INZ(0)
D* High offset of bulk data
D RMDO2         105     108I 0 INZ(0)

```

## Déclaration RPG

### IBM i MQRR (enregistrement de réponse) sur IBM i

La structure MQRR est utilisée pour recevoir le code achèvement et le code anomalie résultant de l'opération d'ouverture ou d'insertion pour une file d'attente de destination unique, lorsque la destination est une liste de distribution.

## Présentation

**Objectif:** MQRR est une structure de sortie pour les appels MQOPEN, MQPUT et MQPUT1 .

**Jeu de caractères et codage:** les données de MQRR doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par ENNAT. Toutefois, si l'application s'exécute en tant que client IBM MQ , la structure doit être dans le jeu de caractères et le codage du client.

**Syntaxe:** en fournissant un tableau de ces structures sur les appels MQOPEN et MQPUT ou sur l'appel MQPUT1 , il est possible de déterminer les codes achèvement et les codes anomalie de toutes les files d'attente d'une liste de distribution lorsque le résultat de l'appel est mixte, c'est-à-dire lorsque l'appel aboutit pour certaines files d'attente de la liste mais échoue pour d'autres. Le code anomalie RC2136 de l'appel indique que les enregistrements de réponse (s'ils sont fournis par l'application) ont été définis par le gestionnaire de files d'attente.

- «Zones», à la page [1265](#)
- «Valeurs initiales», à la page [1266](#)
- «Déclaration RPG», à la page [1266](#)

## Zones

La structure MQRR contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique:**

### RRCC (entier signé à 10 chiffres)

Code achèvement de la file d'attente.

Il s'agit du code achèvement résultant de l'opération d'ouverture ou d'insertion pour la file d'attente avec le nom indiqué par l'élément correspondant dans le tableau des structures MQOR fournies sur l'appel MQOPEN ou MQPUT1 .

Il s'agit toujours d'une zone de sortie. La valeur initiale de ce champ est CCOK.

### RRREA (entier signé à 10 chiffres)

Code raison de la file d'attente.

Il s'agit du code anomalie résultant de l'opération d'ouverture ou d'insertion pour la file d'attente avec le nom qui a été indiqué par l'élément correspondant dans le tableau des structures MQOR fournies sur l'appel MQOPEN ou MQPUT1 .

Il s'agit toujours d'une zone de sortie. La valeur initiale de cette zone est RCNONE.

## Valeurs initiales

Tableau 725. Zones dans MQRR		
Nom de zone	Nom de la constante	Valeur de la constante
RRCC	CCOK	0
RRREA	RCNONE	0

## Déclaration RPG

```

D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQRR Structure
D*
D* Completion code for queue
D RRCC                1          4I 0 INZ(0)
D* Reason code for queue
D RRREA               5          8I 0 INZ(0)

```

## MQSCO (options de configuration TLS) sous IBM i

La structure MQSCO (avec les zones TLS dans la structure MQCD) permet à une application s'exécutant en tant que IBM MQ MQI client de spécifier des options de configuration qui contrôlent l'utilisation de TLS pour la connexion client lorsque le protocole de canal est TCP/IP.

## Présentation

**Objectif:** La structure est un paramètre d'entrée dans l'appel MQCONN.

Si le protocole de canal du canal client n'est pas TCP/IP, la structure MQSCO est ignorée.

**Jeu de caractères et codage:** les données de MQSCO doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par ENNAT.

- [«Zones», à la page 1266](#)
- [«Valeurs initiales», à la page 1271](#)
- [«Déclaration RPG», à la page 1272](#)

## Zones

La structure MQSCO contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique:**

### SCAIC (entier signé à 10 chiffres)

Il s'agit du nombre d'enregistrements d'informations d'authentification (MQAIR) traités par les zones SCAIP ou SCAIO . Pour plus d'informations, voir [«MQAIR \(enregistrement d'informations d'authentification\) sur IBM i», à la page 1062](#). La valeur doit être supérieure ou égale à zéro. Si la valeur n'est pas valide, l'appel échoue avec le code anomalie RC2383.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

### **SCAIO (entier signé à 10 chiffres)**

Décalage en octets du premier enregistrement d'informations d'authentification à partir du début de la structure MQSCO. Le décalage peut être positif ou négatif. La zone est ignorée si *SCAIC* a la valeur zéro.

Vous pouvez utiliser *SCAIO* ou *SCAIP* pour spécifier les enregistrements MQAIR, mais pas les deux ; voir la description de la zone *SCAIP* pour plus de détails.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

### **SCAIP (entier signé à 10 chiffres)**

Il s'agit de l'adresse du premier enregistrement d'informations d'authentification. La zone est ignorée si *SCAIC* a la valeur zéro.

Vous pouvez fournir le tableau d'enregistrements MQAIR de l'une des deux manières suivantes:

- A l'aide de la zone de pointeur *SCAIP*

Dans ce cas, l'application peut déclarer un tableau d'enregistrements MQAIR distinct de la structure MQSCO et définir *SCAIP* sur l'adresse du tableau.

Envisagez d'utiliser *SCAIP* pour les langages de programmation qui prennent en charge le type de données de pointeur d'une manière portable dans différents environnements (par exemple, le langage de programmation C).

- En utilisant la zone de décalage *SCAIO*

Dans ce cas, l'application doit déclarer une structure composée contenant un MQSCO suivi du tableau d'enregistrements MQAIR et définir *SCAIO* sur le décalage du premier enregistrement du tableau à partir du début de la structure MQSCO. Vérifiez que cette valeur est correcte et qu'elle peut être prise en compte dans un MQLONG (le langage de programmation le plus restrictif est COBOL, pour lequel la plage valide est comprise entre -999 999 999 et +999 999 999).

Envisagez d'utiliser *SCAIO* pour les langages de programmation qui ne prennent pas en charge le type de données de pointeur ou qui implémentent le type de données de pointeur d'une manière qui n'est pas portable dans des environnements différents (par exemple, le langage de programmation COBOL).

Quelle que soit la technique choisie, une seule des options *SCAIP* et *SCAIO* peut être utilisée ; l'appel échoue avec le code anomalie RC2384 si les deux sont différents de zéro.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire.

**Remarque :** Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

### **SCCERLBL (entier signé à 10 chiffres)**

Cette zone fournit des détails sur le label de certificat utilisé.

IBM MQ initialise la valeur de la zone SCCERLBL sous forme de blancs. Entrez la valeur requise ou acceptez la valeur par défaut.

`ibmwebsphereuser_id` est une valeur valide pour cette zone pour toutes les versions du produit, et pour les versions de MQSCO inférieures à 5.0, il s'agit de la seule valeur valide. Par conséquent, la valeur de cette zone est interprétée lors de l'exécution et modifiée si nécessaire. Si vous indiquez une version de MQSCO inférieure à 5.0 ou que vous acceptez la valeur par défaut des blancs pour la zone SCCERLBL, le système utilise la valeur `ibmwebsphereuser_id`.

Il s'agit d'une zone d'entrée.

### **SCCERTVPOL (entier signé à 10 chiffres)**

Cette zone indique le type de règle de validation de certificat utilisé. La zone peut être définie sur l'une des valeurs suivantes:

### **MQ\_CERT\_VAL\_POLICY\_ANY**

Appliquez chacune des règles de validation de certificat prises en charge par la bibliothèque de sockets sécurisés. Acceptez la chaîne de certificats si l'une des politiques considère que la chaîne de certificats est valide.

### **MQ\_CERT\_VAL\_POLICY\_RFC5280**

Appliquez uniquement la règle de validation de certificat RFC5280 . Ce paramètre fournit une validation plus stricte que le paramètre ANY, mais rejette certains certificats numériques plus anciens.

La valeur initiale de cette zone est MQ\_CERT\_VAL\_POLICY\_ANY

### **SCCH (entier signé à 10 chiffres)**

Cette zone fournit des détails de configuration pour le matériel cryptographique connecté au système client.

Définissez la zone sur une chaîne au format suivant, ou laissez-la vide ou null:

```
GSK_PKCS11=the PKCS #11 driver path and file name;the PKCS #11 token label;the PKCS #11 token password;symmetric cipher setting>;
```

Pour utiliser du matériel cryptographique conforme à l'interface PKCS11 , par exemple IBM 4960 ou IBM 4963, indiquez le chemin du pilote PKCS11 , le libellé de jeton PKCS11 et les chaînes de mot de passe de jeton PKCS11 , chacune se terminant par un point-virgule.

Le chemin du pilote PKCS #11 est un chemin d'accès absolu à la bibliothèque partagée prenant en charge la carte PKCS #11 . Le nom de fichier du pilote PKCS #11 est le nom de la bibliothèque partagée. Voici un exemple de la valeur requise pour le chemin d'accès et le nom de fichier PKCS #11 :

```
/usr/lib/pkcs11/PKCS11_API.so
```

Le libellé de jeton PKCS #11 doit être entièrement en minuscules. Si vous avez configuré votre matériel avec un libellé de jeton en casse mixte ou en majuscule, reconfigurez-le avec ce libellé en minuscule.

Si aucune configuration de matériel de cryptographie n'est requise, définissez la zone sur une valeur vide ou null.

Si la valeur est inférieure à la longueur de la zone, terminez la valeur par un caractère NULL ou remplissez la zone avec des blancs. Si la valeur n'est pas valide ou entraîne un échec lors de l'utilisation de la configuration du matériel de cryptographie, l'appel échoue avec le code anomalie RC2382.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par LNSSCH. La valeur initiale de cette zone est des caractères blancs.

### **SCEPSUITEB (entier signé à 10 chiffres)**

Cette zone indique si la cryptographie compatible Suite B est utilisée et quel niveau de puissance est utilisé. La valeur peut être une ou plusieurs des valeurs suivantes:

- SCEPSUITEB0  
La cryptographie conforme à la suite B n'est pas utilisée.
- SCEPSUITEB1  
La sécurité de la suite B 128 bits est utilisée.
- SCEPSUITEB2  
La sécurité de la suite B 192 bits est utilisée.

**Remarque :** L'utilisation de SCEPSUITEB0 avec une autre valeur dans cette zone n'est pas valide.

## SCFR (entier signé à 10 chiffres)

IBM MQ peut être configuré avec du matériel de cryptographie de sorte que les modules de cryptographie utilisés soient ceux fournis par le produit matériel ; ils peuvent être certifiés FIPS à un niveau particulier en fonction du produit matériel de cryptographie utilisé.

Utilisez cette zone pour indiquer que seuls les algorithmes certifiés FIPS sont utilisés si la cryptographie est fournie dans le logiciel fourni par IBM MQ.

Lorsque IBM MQ est installé, une implémentation de la cryptographie TLS est également installée, qui fournit des modules certifiés FIPS.

Les valeurs possibles sont les suivantes:

### **MQSSL\_FIPS\_NO**

Il s'agit de la valeur par défaut. Lorsque cette valeur est définie:


- Tout CipherSpec pris en charge sur une plateforme particulière peut être utilisé.
- En cas d'exécution sans utilisation de matériel de cryptographie, les CipherSpecs suivants s'exécutent à l'aide de la cryptographie certifiée FIPS 140-2 sur les plateformes IBM MQ :
  - TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA
  - TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA
  - TLS\_RSA\_WITH\_AES\_256\_CBC\_SHA

### **MQSSL\_FIPS\_YES**

Lorsque cette valeur est définie, sauf si vous utilisez du matériel de cryptographie pour effectuer la cryptographie, vous pouvez être certain que

- Seuls les algorithmes de cryptographie certifiés FIPS peuvent être utilisés dans le CipherSpec s'appliquant à cette connexion client.
- Les connexions de canal TLS entrantes et sortantes aboutissent uniquement si l'une des spécifications de chiffrement suivantes est utilisée:
  - TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA
  - TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA
  - TLS\_RSA\_WITH\_AES\_256\_CBC\_SHA

### **Remarques :**

1.  CipherSpec TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA est obsolète.
2. Dans la mesure du possible, si des CipherSpecs FIPS uniquement sont configurés, le client MQI rejette les connexions qui spécifient un CipherSpec withRC2393. IBM MQ ne garantit pas le rejet de toutes ces connexions et il est de votre responsabilité de déterminer si votre configuration IBM MQ est conforme à la norme FIPS.

► V 9.3.0

► V 9.3.0

## **SCKEYPWL (entier signé à 10 chiffres)**

Longueur de la phrase passe du référentiel de clés TLS.

La longueur maximale de la phrase passe du référentiel de clés est de 128 caractères. Si la phrase passe du référentiel de clés est supérieure à la longueur maximale autorisée, la connexion échoue avec RC2381.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

► V 9.3.0

► V 9.3.0

## **SCKEYPWO (entier signé à 10 chiffres)**

Décalage en octets de la phrase passe du référentiel de clés TLS. Le décalage peut être positif ou négatif.

Vous pouvez utiliser SCKEYPWO ou SCKEYPWP pour spécifier la phrase passe du référentiel de clés, mais pas les deux. Pour plus d'informations, voir la description de la zone SCKEYPWP .

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

### **V 9.3.0** **V 9.3.0** **SCKEYPWP (pointeur)**

Il s'agit de l'adresse de la phrase passe du référentiel de clés TLS.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null.

La phrase passe du référentiel de clés peut être spécifiée sous la forme d'une chaîne de texte en clair ou d'une phrase passe qui a été chiffrée à l'aide de l'utilitaire **runmqicred**.

La phrase passe du référentiel de clés spécifiée à l'aide de cette zone remplace toute phrase passe du référentiel de clés spécifiée à l'aide de la variable d'environnement MQKEYRPWD ou de la propriété SSLKeyRepositoryPassword dans la section SSL du fichier de configuration du client.

Vous pouvez utiliser SCKEYPW0 ou SCKEYPWP pour spécifier la phrase passe du référentiel de clés, mais pas les deux.

### **SCKR (entier signé à 10 chiffres)**

Cette zone indique l'emplacement du fichier de la base de données de clés dans lequel les clés et les certificats sont stockés. **V 9.3.0** **V 9.3.0** Si le suffixe de fichier n'est pas spécifié, le suffixe .kdb est ajouté automatiquement.

Chaque fichier de base de données de clés peut être associé à un *fichier de mot de passe secret*. Contient les mots de passe chiffrés qui sont utilisés pour autoriser l'accès par programmation à la base de données de clés. Le fichier de mot de passe secret doit résider dans le même répertoire et avoir le même radical de fichier que la base de données de clés, et doit se terminer par le suffixe .sth.

Par exemple, si le fichier de la base de données de clés est /xxx/yyy/key.kdb, le fichier de dissimulation de mot de passe doit être /xxx/yyy/key.sth, où xxx et yyy représentent les noms de répertoire.

**V 9.3.0** **V 9.3.0** Le mot de passe de la base de données de clés peut également être spécifié à l'aide des zones SCKEYPWP ou SCKEYPW0.

Si la valeur est inférieure à la longueur de la zone, terminez la valeur par un caractère NULL ou remplissez la zone avec des blancs. La valeur n'est pas vérifiée ; en cas d'erreur lors de l'accès au référentiel de clés, l'appel échoue avec le code anomalie RC2381.

Pour exécuter une connexion TLS à partir d'un IBM MQ MQI client, définissez SCKR sur un nom de fichier de base de données de clés valide.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par LNSSKR. La valeur initiale de cette zone est un caractère blanc.

### **SCSID (entier signé à 10 chiffres)**

Il s'agit de l'identificateur de structure ; la valeur doit être:

#### **SCSIDV**

Identificateur de la structure des options de configuration TLS.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est SCSIDV.

### **SCVER (entier signé à 10 chiffres)**

Il s'agit du numéro de version de la structure ; la valeur doit être:

#### **SCVER1**

Structure des options de configuration TLS Version-1.

#### **SCVER2**

Structure des options de configuration TLS Version-2.

#### **SCVER3**

Structure des options de configuration TLS Version-3.

**SCVER4**

Structure des options de configuration TLS Version-4 .

**SCVER5**

Structure des options de configuration TLS Version-5 .

**V 9.3.0** **V 9.3.0** **SCVER6**

Structure des options de configuration TLS Version-6 .

La constante suivante indique le numéro de version de la version en cours:

**SCVERC**

Version actuelle de la structure des options de configuration TLS.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est SCVER1.

**Valeurs initiales**

Tableau 726. Zones dans MQSCO

Nom de zone	Nom de la constante	Valeur de la constante
SCSID	SCSIDV	' SC0↵ '
SCVER	SCVER1	1
SCKR	Aucun	Chaîne nulle ou blancs
SCCH	Aucun	Chaîne nulle ou blancs
SCAIC	Aucun	0
SCAIO	Aucun	0
SCAIP	Aucun	Pointeur null ou octets null
SCKRC	Aucun	Pointeur null ou octets null
SCFR	Aucun	Pointeur null ou octets null
SCEPSUITEB	Aucun	Pointeur null ou octets null
SCCERTVPOL	Aucun	Pointeur null ou octets null
SCCERLBL	Aucun	Pointeur null ou octets null
<b>V 9.3.0</b> <b>V 9.3.0</b> SCKEYPWP	Aucun	Pointeur null ou octets null
<b>V 9.3.0</b> <b>V 9.3.0</b> SCKEYPWO	Aucun	0
<b>V 9.3.0</b> <b>V 9.3.0</b> SCKEYPWL	Aucun	0

Tableau 726. Zones dans MQSCO (suite)

Nom de zone	Nom de la constante	Valeur de la constante
<b>Remarques :</b>		
1. Le symbole $\bar{\phantom{x}}$ représente un caractère blanc unique.		
2. Pour connaître les options SCEPSUITEB , voir «Déclaration RPG», à la page 1272 .		

## Déclaration RPG

```

D*..1....:....2.....3.....4.....5.....6.....7..
D* MQSCO Structure
D*
D* Structure identifieur
D SCSID          1      4    INZ('SCO ')
D* Structure version number
D SCVER          5      8I 0 INZ(1)
D* Location of TLS key repository
D SCKR           9     264    INZ
D* Cryptographic hardware configuration string
D SCCH           265    520    INZ
D* Number of MQAIR records present
D SCAIC          521    524I 0 INZ(0)
D* Offset of first MQAIR record from start of MQSCO structure
D SCAIO          525    528I 0 INZ(0)
D* Address of first MQAIR record
D SCAIP          529    544*   INZ(*NULL)
D* Ver:1 **
D* Number of unencrypted bytes sent/received before secret key is
D* reset
D SCKRC          545    548I 0 INZ(0)
D* Using FIPS-certified algorithms
D SCFR           549    552I 0 INZ(0)
D* Ver:2 **
* Use only Suite B cryptographic algorithms
D SCEPSUITEB0
D SCEPSUITEB1    553    556I 0 INZ(1)
D SCEPSUITEB2    557    560I 0 INZ(0)
D SCEPSUITEB3    561    564I 0 INZ(0)
D SCEPSUITEB4    565    568I 0 INZ(0)
D SCEPSUITEB     10I 0 DIM(4) OVERLAY(SCEPSUITEB0)
D* Ver:3 **
D* Certificate validation policy
D SCCERTVPOL     569    572I 0 INZ(0)
D* Ver:4 **

```



## MQSD (descripteur d'abonnement) sous IBM i

La structure MQSD permet de spécifier les détails de l'abonnement effectué.

### Présentation

#### Objet

La structure est un paramètre d'entrée-sortie dans l'appel MQSUB.

#### Abonnements gérés

Si une application n'a pas besoin d'utiliser une file d'attente particulière comme destination pour les publications qui correspondent à son abonnement, elle peut utiliser la fonction d'abonnement géré. Si une application choisit d'utiliser un abonnement géré, le gestionnaire de files d'attente informe l'abonné de la destination où les messages publiés sont envoyés, en fournissant un descripteur d'objet comme sortie de l'appel MQSUB. Pour plus d'informations, voir [HOBj \(10-digits signed integer\)-input/output](#).

Lorsque l'abonnement est supprimé, le gestionnaire de files d'attente s'engage également à nettoyer les messages qui n'ont pas été extraits de la destination gérée, dans les cas suivants:



- Lorsque l'abonnement est supprimé-à l'aide de MQCLOSE avec CORMSB-et que le Hobj géré est fermé.
- Par des moyens implicites lorsque la connexion est perdue à une application à l'aide d'un abonnement non durable (SONDUR)
- Par expiration lorsqu'un abonnement est supprimé car il est arrivé à expiration et que le Hobj géré est fermé.

Vous devez utiliser des abonnements gérés avec des abonnements non durables, afin que le nettoyage puisse se produire et que les messages des abonnements non durables fermés n'occupent pas d'espace dans votre gestionnaire de files d'attente. Les abonnements durables peuvent également utiliser des destinations gérées.

### **Jeu de caractères et codage**

Les données de MQSD doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par ENNAT. Toutefois, si l'application s'exécute en tant que client IBM MQ , la structure doit être dans le jeu de caractères et le codage du client.

- «Zones», à la page [1273](#)
- «Valeurs initiales», à la page [1286](#)
- «Déclaration RPG», à la page [1286](#)

### **Zones**

La structure MQSD contient les zones suivantes ; les zones sont décrites par ordre alphabétique:

#### **SDAID (chaîne de caractères de 32 octets)**

Cette valeur se trouve dans la zone *MDAID* du descripteur de message (MQMD) de tous les messages de publication correspondant à cet abonnement. *SDAID* fait partie du contexte d'identité du message. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#).

Pour plus d'informations sur *MDAID* , voir [MDAID](#).

Si l'option SOSETI n'est pas spécifiée, le paramètre *MDAID* défini dans chaque message publié pour cet abonnement est vide, en tant qu'informations de contexte par défaut.

Si l'option SOSETI est spécifiée, le fichier *SDAID* est généré par l'utilisateur et cette zone est une zone d'entrée qui contient le fichier *MDAID* à définir dans chaque publication pour cet abonnement.

La longueur de cette zone est indiquée par LNAIDD. La valeur initiale de cette zone est de 32 caractères blancs.

Si vous modifiez un abonnement existant à l'aide de l'option SOALT, vous pouvez modifier le *SDAID* de tout message de publication futur.

En cas de retour d'un appel MQSUB à l'aide de SORES, cette zone est définie sur le *MDAID* en cours utilisé pour l'abonnement.

#### **SDACC (chaîne de caractères de 32 octets)**

Cette valeur se trouve dans la zone *MDACC* du descripteur de message (MQMD) de tous les messages de publication correspondant à cet abonnement. *MDACC* fait partie du contexte d'identité du message. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#).

Pour plus d'informations sur *MDACC* , voir [MDACC](#).

Vous pouvez utiliser la valeur spéciale suivante pour la zone *SDACC* :

#### **ACNONE**

Aucun jeton de comptabilité n'est spécifié.

La valeur est zéro binaire pour la longueur de la zone.

Si l'option SOSETI n'est pas spécifiée, le jeton de comptabilité est généré par le gestionnaire de files d'attente en tant qu'informations de contexte par défaut et cette zone est une zone de sortie qui contient le *MDACC* défini dans chaque message publié pour cet abonnement.

Si l'option SOSETI est spécifiée, le jeton de comptabilité est généré par l'utilisateur et cette zone est une zone d'entrée qui contient le *MDACC* à définir dans chaque publication pour cet abonnement.

La longueur de cette zone est indiquée par LNACCT. La valeur initiale de cette zone est ACNONE.

Si vous modifiez un abonnement existant à l'aide de l'option SOALT, la valeur de *MDACC* dans les messages de publication ultérieurs peut être modifiée.

En cas de retour d'un appel MQSUB à l'aide de SORES, cette zone est définie sur le *MDACC* en cours utilisé pour l'abonnement.

### **SDASI (chaîne de bits de 40 octets)**

Il s'agit d'un identificateur de sécurité qui est transmis avec *SDAU* au service d'autorisation pour permettre l'exécution des vérifications d'autorisation appropriées.

*SDASI* est utilisé uniquement si SOALTU est spécifié et que la zone *SDAU* n'est pas entièrement vide jusqu'au premier caractère null ou à la fin de la zone.

En cas de retour d'un appel MQSUB à l'aide de SORES, cette zone n'est pas modifiée.

Pour plus d'informations, voir la description de *ODASI* dans le type de données MQOD.

### **SDAU (chaîne de caractères de 12 octets)**

Si vous spécifiez SOALTU, cette zone contient un autre identificateur utilisateur qui est utilisé pour vérifier l'autorisation de l'abonnement et de la sortie dans la file d'attente de destination (spécifiée dans le paramètre **Hobj** de l'appel MQSUB), à la place de l'identificateur utilisateur sous lequel l'application est en cours d'exécution.

Si l'opération aboutit, l'ID utilisateur indiqué dans cette zone est enregistré en tant qu'ID utilisateur propriétaire de l'abonnement à la place de l'ID utilisateur sous lequel l'application est en cours d'exécution.

Si SOALTU est spécifié et que cette zone est entièrement vide jusqu'au premier caractère null ou jusqu'à la fin de la zone, l'abonnement peut aboutir uniquement si aucune autorisation utilisateur n'est nécessaire pour s'abonner à cette rubrique avec les options spécifiées ou la file d'attente de destination pour la sortie.

Si SOALTU n'est pas spécifié, cette zone est ignorée.

En cas de retour d'un appel MQSUB à l'aide de SORES, cette zone n'est pas modifiée.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par LNUID. La valeur initiale de cette zone est de 12 caractères blancs.

### **SDCID (chaîne de 24 octets)**

Toutes les publications envoyées pour correspondre à cet abonnement contiennent cet identificateur de corrélation dans le descripteur de message. Si plusieurs abonnements utilisent la même file d'attente pour obtenir leurs publications, l'utilisation de MQGET par ID de corrélation permet d'obtenir uniquement les publications d'un abonnement spécifique. Cet identificateur de corrélation peut être généré par le gestionnaire de files d'attente ou par l'utilisateur.

Si l'option SOSCID n'est pas spécifiée, l'identificateur de corrélation est généré par le gestionnaire de files d'attente et cette zone est une zone de sortie qui contient l'identificateur de corrélation défini dans chaque message publié pour cet abonnement.

Si l'option SOSCID est spécifiée, l'identificateur de corrélation est généré par l'utilisateur et cette zone est une zone d'entrée qui contient l'identificateur de corrélation à définir dans chaque publication pour cet abonnement. Dans ce cas, si le champ contient CINONE, l'identificateur de corrélation qui est défini dans chaque message publié pour cet abonnement est l'identificateur de corrélation qui a été créé par l'insertion d'origine du message.

Si l'option SOGRP est spécifiée et que l'identificateur de corrélation spécifié est identique à un abonnement groupé existant utilisant la même file d'attente et une chaîne de rubrique se chevauchant, seul l'abonnement le plus significatif du groupe est fourni avec une copie de la publication.

La longueur de cette zone est indiquée par LNCID. La valeur initiale de cette zone est CINONE.

Si vous modifiez un abonnement existant à l'aide de l'option SOALT et que cette zone est une zone d'entrée, l'ID de corrélation de l'abonnement peut être modifié, sauf si l'abonnement a été créé à l'aide de l'option SOGRP .

En cas de retour d'un appel MQSUB à l'aide de SORES, cette zone est définie sur l'ID de corrélation en cours de l'abonnement.

### **SDEXP (entier signé à 10 chiffres)**

Il s'agit du temps exprimé en dixièmes de seconde après lequel l'abonnement expire. Aucune autre publication ne correspondra à cet abonnement une fois cet intervalle écoulé. Elle est également utilisée comme valeur dans la zone *MDEXP* du MQMD des publications envoyées à cet abonné.

La valeur spéciale suivante est reconnue:

#### **EIULIM**

L'abonnement a un délai d'expiration illimité.

Si vous modifiez un abonnement existant à l'aide de l'option SOALT , l'expiration de l'abonnement peut être modifiée.

En cas de retour d'un appel MQSUB à l'aide de l'option SORES , cette zone est définie sur l'expiration d'origine de l'abonnement et non sur l'heure d'expiration restante.

### **SDON (chaîne de caractères de 48 octets)**

Il s'agit du nom de l'objet de rubrique tel qu'il est défini sur le gestionnaire de files d'attente local.

Le nom peut contenir les caractères suivants:

- Caractères alphabétiques en majuscules (A à Z)
- Caractères alphabétiques minuscules (a à z)
- Chiffres (0 à 9)
- Point (.), barre oblique (/), trait de soulignement (\_), pourcentage (%)

Le nom ne doit pas contenir d'espaces de début ou d'espaces imbriqués, mais peut contenir des espaces de fin. Utilisez un caractère null pour indiquer la fin des données significatives dans le nom ; la valeur null et les caractères qui la suivent sont traités comme des blancs. Les restrictions suivantes s'appliquent :

- Sur les systèmes qui utilisent EBCDIC Katakana, les caractères minuscules ne peuvent pas être utilisés.
- Les noms contenant des caractères minuscules, des barres obliques ou des pourcentages doivent être placés entre guillemets lorsqu'ils sont spécifiés dans les commandes. Ces guillemets ne doivent pas être indiqués pour les noms qui apparaissent en tant que zones dans les structures ou en tant que paramètres dans les appels.

*SDON* est utilisé pour former le nom de rubrique complet.

Le nom complet de la rubrique peut être généré à partir de deux zones différentes: *SDON* et *SDOS*. Pour plus de détails sur l'utilisation de ces deux zones, voir [Combinaison de chaînes de rubrique](#).

En cas de retour d'un appel MQSUB à l'aide de l'option SORES , cette zone est inchangée.

La longueur de cette zone est indiquée par LNTOPN. La valeur initiale de cette zone est de 48 caractères blancs.

Si vous modifiez un abonnement existant à l'aide de l'option SDALT, le nom de l'objet de rubrique auquel vous êtes abonné ne peut pas être modifié. Cette zone et *SDOS* peuvent être omises. S'ils

sont fournis, ils doivent être résolus avec le même nom de rubrique complet ou l'appel échoue avec RC2510 .

### SDOPT (entier signé à 10 chiffres)

Vous devez spécifier au moins l'une des options suivantes:

- SOALT
- SORES (Sorces)
- SOCRT

Les valeurs peuvent être ajoutées. N'ajoutez pas la même constante plus d'une fois. Le tableau montre comment combiner ces options: les combinaisons qui ne sont pas valides sont notées ; les autres combinaisons sont valides.

#### Options d'accès ou de création

Les options d'accès et de création déterminent si un abonnement est créé ou si un abonnement existant est renvoyé ou modifié. Vous devez spécifier au moins l'une de ces options. Le tableau affiche les combinaisons valides d'options d'accès ou de création.

<i>Tableau 727. Combinaisons valides d'options d'accès et de création</i>	
<b>Combinaison d'options</b>	<b>Remarques</b>
SOCRT	Crée un abonnement s'il n'en existe pas ; échoue s'il existe.
SORES	Reprend un abonnement existant, échoue s'il n'existe pas d'abonnement.
SOCRT + SORES	Crée un abonnement s'il n'en existe pas et en reprend un correspondant, s'il existe. Combinaison utile si elle est utilisée dans une application qui peut être exécutée plusieurs fois.
SORES + SOALT (voir note)	Reprend un abonnement existant, en modifiant les zones pour qu'elles correspondent à celles spécifiées dans le MQSD, échoue s'il n'existe pas d'abonnement.
SOCRT + SOALT (voir remarque)	Crée un abonnement s'il n'en existe pas et en reprend un correspondant, s'il existe, en modifiant les zones pour qu'elles correspondent à celles spécifiées dans le MQSD. Combinaison utile si elle est utilisée dans une application qui souhaite s'assurer que son abonnement est dans un certain état avant de continuer.

#### Remarque :

Les options spécifiant SOALT peuvent également spécifier SORES, mais cette combinaison n'a aucun effet supplémentaire sur la spécification de SOALT seule. SOALT implique SORES, car l'appel de MQSUB pour modifier un abonnement implique que les abonnements sont également repris. Le contraire n'est pas vrai, cependant: la reprise d'un abonnement n'implique pas qu'il doit être modifié.

#### SOCRT

Créez un abonnement pour la rubrique spécifiée. Si un abonnement utilisant le même SDSN existe, l'appel échoue avec RC2432 . Cet échec peut être évité en combinant l'option SOCRT avec SORES. Le SDSN n'est pas toujours nécessaire. Pour plus de détails, voir la description de cette zone.

La combinaison de SOCRT avec SORES vérifie d'abord s'il existe un abonnement existant pour le SDSNspécifié et s'il existe un descripteur pour cet abonnement préexistant ; mais s'il n'existe pas

d'abonnement existant, un nouvel abonnement est créé à l'aide de toutes les zones fournies dans le MQSD.

SOCRT peut également être combiné à SOALT pour obtenir un effet similaire (voir les détails sur SOALT plus loin dans cette rubrique).

## **SORES**

Renvoie un descripteur à un abonnement préexistant qui correspond à ceux spécifiés par *SDSN*. Aucune modification n'est apportée aux attributs d'abonnement correspondants et ils sont renvoyés en sortie dans la structure MQSD. La plupart des contenus du MQSD ne sont pas utilisés: les zones utilisées sont *SDSID*, *SDVER*, *SDOPT*, *SDAID* et *SDASIE* et *SDSN*.

L'appel échoue avec le code anomalie RC2428 s'il n'existe pas d'abonnement correspondant au nom d'abonnement complet. Cet échec peut être évité en combinant l'option SOCRT avec SORES. Pour plus de détails sur SOCRT, voir [SOCRT](#).

L'ID utilisateur de l'abonnement est l'ID utilisateur qui a créé l'abonnement, ou s'il a été modifié ultérieurement par un autre ID utilisateur, il s'agit de l'ID utilisateur de la modification la plus récente ayant abouti. Si un *SDAID* est utilisé et que l'utilisation d'autres ID utilisateur est autorisée pour cet utilisateur, *SDAID* est enregistré en tant qu'ID utilisateur qui a créé l'abonnement à la place de l'ID utilisateur sous lequel l'abonnement a été effectué.

L'ID utilisateur qui a créé l'abonnement est enregistré sous la forme *SDAU* si cette zone est utilisée et l'utilisation d'autres ID utilisateur est autorisée pour cet utilisateur.

S'il existe un abonnement correspondant qui a été créé sans l'option SOAUID et que l'ID utilisateur de l'abonnement est différent de celui de l'application demandant un descripteur pour l'abonnement, l'appel échoue avec le code anomalie RC2434 .

Si un abonnement correspondant existe et est actuellement utilisé par une autre application, l'appel échoue avec le code anomalie RC2429 . S'il est actuellement utilisé par la même connexion, l'appel n'échoue pas et un descripteur de l'abonnement est renvoyé.

Si l'abonnement nommé dans SubName n'est pas un abonnement valide à reprendre ou à modifier à partir d'une application, l'appel échoue avec RC2523 .

SORES est implicite dans SOALT et ne doit donc pas être combiné avec cette option. Cependant, il ne s'agit pas d'une erreur si ces deux options sont combinées.

## **SOALT**

Renvoie un descripteur à un abonnement préexistant dont le nom d'abonnement complet correspond à ceux spécifiés dans *SDSN*. Tous les attributs de l'abonnement qui sont différents de ceux spécifiés dans le MQSD sont modifiés dans l'abonnement, sauf si la modification est interdite pour cet attribut. Les détails sont indiqués dans la description de chaque attribut et sont résumés dans le tableau suivant. Si vous tentez de modifier un attribut qui ne peut pas être modifié, l'appel échoue avec le code anomalie indiqué dans le tableau suivant.

L'appel échoue avec le code anomalie RC2428 s'il n'existe pas d'abonnement correspondant au nom d'abonnement complet. Cet échec peut être évité en combinant l'option SOCRT avec SOALT.

La combinaison de SOCRT avec SOALT vérifie d'abord s'il existe un abonnement existant pour le nom d'abonnement complet spécifié, et si un descripteur est renvoyé à cet abonnement préexistant avec des modifications apportées comme précédemment détaillé ; mais s'il n'existe pas d'abonnement existant, un nouvel abonnement est créé à l'aide de toutes les zones fournies dans le MQSD.

L'ID utilisateur de l'abonnement est l'ID utilisateur qui a créé l'abonnement, ou s'il a été modifié ultérieurement par un autre ID utilisateur, il s'agit de l'ID utilisateur de la modification ayant abouti la plus récente. Si *SDAU* est utilisé (et que l'utilisation d'ID utilisateur de remplacement est autorisée pour cet utilisateur), l'ID utilisateur de remplacement est enregistré en tant qu'ID utilisateur qui a créé l'abonnement à la place de l'ID utilisateur sous lequel l'abonnement a été effectué.

S'il existe un abonnement correspondant qui a été créé sans l'option SOAUID et que l'ID utilisateur de l'abonnement est différent de celui de l'application demandant un descripteur pour l'abonnement, l'appel échoue avec le code anomalie RC2434 .

Si un abonnement correspondant existe et est actuellement utilisé par une autre application, l'appel échoue avec RC2429 . S'il est actuellement utilisé par la même connexion, l'appel n'échoue pas et un descripteur de l'abonnement est renvoyé.

Si l'abonnement nommé dans SubName n'est pas un abonnement valide à reprendre ou à modifier à partir d'une application, l'appel échoue avec RC2523 .

Les tableaux suivants présentent les attributs d'abonnement qui peuvent être modifiés par SOALT.

<i>Tableau 728. Attributs dans MQSD et MQSUB pouvant être modifiés</i>			
<b>Descripteur de type de données ou appel de fonction</b>	<b>Nom de zone</b>	<b>Cet attribut peut-il être modifié à l'aide de SOALT?</b>	<b>Code raison</b>
Disque MQSD	Options de durabilité	Non	RC2509
Disque MQSD	Options de destination	Oui	Aucun
Disque MQSD	Options d'enregistrement	Oui (voir la remarque <u>1</u> )	RC2515 si vous essayez de modifier SOGRP
Disque MQSD	Options de publication	Oui (voir la remarque <u>2</u> )	Aucun
Disque MQSD	Options de caractère générique	Non	RC2510
Disque MQSD	Autres options	Non (voir la remarque <u>3</u> )	Aucun
Disque MQSD	ObjectName	Non	RC2510
Disque MQSD	SDAU	Non (voir la remarque <u>4</u> )	Aucun
Disque MQSD	SDASI	Non (voir la remarque <u>4</u> )	Aucun
Disque MQSD	SDEXP	Oui	Aucun
Disque MQSD	SDOS	Non	RC2510
Disque MQSD	SDSN	Non (voir la remarque <u>5</u> )	Aucun
Disque MQSD	SDSUD	Oui	Aucun
Disque MQSD	SDCID	Oui (voir la remarque <u>6</u> )	RC2515 dans un abonnement groupé
Disque MQSD	SDPRI	Oui	Aucun
Disque MQSD	SDACC	Oui	Aucun
Disque MQSD	SDAID	Oui	Aucun
Disque MQSD	SDSL	Non	RC2512
MQSUB	HOBJ	Oui (voir la remarque <u>6</u> )	RC2515 dans un abonnement groupé

**Remarques :**

1. SOGRP ne peut pas être modifié.
2. SONEWP ne peut pas être modifié car il ne fait pas partie de l'abonnement
3. Ces options ne font pas partie de l'abonnement
4. Cet attribut ne fait pas partie de l'abonnement

5. Cet attribut correspond à l'identité de l'abonnement en cours de modification

6. Modifiable sauf lorsqu'il fait partie d'un sous-groupe ( SOGRP )

**Options de durabilité:** Les options suivantes contrôlent la durabilité de l'abonnement. Vous ne pouvez spécifier qu'une seule de ces options. Si vous modifiez un abonnement existant à l'aide de l'option SOALT , vous ne pouvez pas modifier la durabilité de l'abonnement. En cas de retour d'un appel MQSUB à l'aide de SORES, l'option de durabilité appropriée est définie.

#### **SODUR**

Demandez que l'abonnement à cette rubrique soit conservé jusqu'à ce qu'il soit explicitement supprimé à l'aide de MQCLOSE avec l'option CORMSB . Si cet abonnement n'est pas supprimé de manière explicite, il sera conservé même après la fermeture de la connexion de cette application au gestionnaire de files d'attente.

Si un abonnement durable est demandé à une rubrique définie comme n'autorisant pas les abonnements durables, l'appel échoue avec RC2436 .

#### **SONDUR**

Demandez que l'abonnement à cette rubrique soit supprimé lorsque la connexion d'application au gestionnaire de files d'attente est fermée, si elle n'a pas déjà été explicitement supprimée. SONDUR est l'opposé de l'option SODUR et est défini pour faciliter la documentation du programme. Il s'agit de la valeur par défaut si aucune n'est spécifiée.

**Options de destination:** Les options suivantes contrôlent la destination à laquelle les publications d'une rubrique à laquelle vous avez souscrit sont envoyées. Si vous modifiez un abonnement existant à l'aide de l'option SOALT, la destination utilisée pour les publications de l'abonnement peut être modifiée. En cas de retour d'un appel MQSUB à l'aide de SORES, cette option est définie si nécessaire.

#### **SOMAN**

Demande que la destination à laquelle les publications sont envoyées soit gérée par le gestionnaire de files d'attente.

Le descripteur d'objet renvoyé dans *HOBJ* représente une file d'attente gérée par le gestionnaire de files d'attente et doit être utilisé avec les appels MQGET, MQCB, MQINQ ou MQCLOSE ultérieurs.

Un descripteur d'objet renvoyé par un appel MQSUB précédent ne peut pas être fourni dans le paramètre **Hobj** lorsque SOMAN n'est pas spécifié.

**Options d'enregistrement:** Les options suivantes contrôlent les détails de l'enregistrement effectué auprès du gestionnaire de files d'attente pour cet abonnement. Si vous modifiez un abonnement existant à l'aide de l'option SOALT , ces options d'enregistrement peuvent être modifiées. En cas de retour d'un appel MQSUB à l'aide de SORES , les options d'enregistrement appropriées sont définies.

#### **SOGRP**

Cet abonnement est regroupé avec d'autres abonnements du même *SDSL* à l'aide de la même file d'attente et en spécifiant le même ID de corrélation de sorte que toute publication dans des rubriques qui entraînerait la fourniture de plusieurs messages de publication au groupe d'abonnements, en raison du chevauchement d'un ensemble de chaînes de rubrique en cours d'utilisation, entraîne la distribution d'un seul message dans la file d'attente. Si cette option n'est pas utilisée, chaque abonnement unique (identifié par *SDSN*) qui correspond est fourni avec une copie de la publication, ce qui peut signifier que plusieurs copies de la publication peuvent être placées dans la file d'attente partagée par un certain nombre d'abonnements.

Seul l'abonnement le plus important du groupe est fourni avec une copie de la publication. L'abonnement le plus significatif est basé sur le nom de rubrique complet jusqu'au point où un caractère générique est trouvé. Si une combinaison de schémas génériques est utilisée dans le groupe, seule la position du caractère générique est importante. Il est conseillé de ne pas combiner différents schémas de caractères génériques dans un groupe d'abonnements partageant la même file d'attente.

Lors de la création d'un abonnement groupé, il doit toujours avoir un *SDSMunique*, mais s'il correspond au nom de rubrique complet d'un abonnement existant dans le groupe, l'appel échoue avec RC2514 .

Si l'abonnement le plus significatif du groupe spécifie également *SONOLC* et qu'il s'agit d'une publication provenant de la même application, aucune publication n'est distribuée dans la file d'attente.

Lors de la modification d'un abonnement effectué avec cette option, les zones qui impliquent le regroupement, *Hobj* sur l'appel *MQSUB* (représentant le nom de la file d'attente et du gestionnaire de files d'attente) et *SDCID* ne peuvent pas être modifiées. La tentative de modification entraîne l'échec de l'appel avec RC2515 .

Cette option doit être combinée avec *SOSCID* avec un *SDCID* qui n'est pas défini sur *CINONE* et qui ne peut pas être combiné avec *SOMAN*.

## **ID SOAU**

Lorsque *SOAUID* est spécifié, l'identité de l'abonné n'est pas limitée à un seul ID utilisateur. Cela permet à tout utilisateur de modifier ou de reprendre l'abonnement lorsqu'il dispose des droits appropriés. Un seul utilisateur peut disposer de l'abonnement à la fois. Une tentative de reprise de l'utilisation d'un abonnement actuellement utilisé par une autre application entraîne l'échec de l'appel avec RC2429 .

Pour ajouter cette option à un abonnement existant, l'appel *MQSUB*, qui utilise *SOALT*, doit provenir du même ID utilisateur que l'abonnement d'origine lui-même.

Si un appel *MQSUB* fait référence à un abonnement existant avec l'ensemble *SOAUID* et que l'ID utilisateur diffère de l'abonnement d'origine, l'appel aboutit uniquement si le nouvel ID utilisateur est autorisé à s'abonner à la rubrique. Une fois l'opération terminée, les futures publications destinées à cet abonné sont placées dans la file d'attente de l'abonné avec le nouvel ID utilisateur défini dans le message de publication.

Ne spécifiez pas à la fois *SOAUID* et *SOFUID*. Si aucun n'est spécifié, la valeur par défaut est *SOFUID*.

## **SOFUID**

Lorsque *SOFUID* est spécifié, l'abonnement peut être modifié ou repris uniquement par le dernier ID utilisateur pour modifier l'abonnement. Si l'abonnement n'a pas été modifié, il s'agit de l'ID utilisateur qui a créé l'abonnement.

Si une instruction *MQSUB* fait référence à un abonnement existant avec *SOAUID* défini et modifie l'abonnement à l'aide de *SOALT* pour utiliser l'ID utilisateur *SOFUID*, l'ID utilisateur de l'abonnement est désormais corrigé avec ce nouvel ID utilisateur. L'appel aboutit uniquement si le nouvel ID utilisateur est autorisé à s'abonner à la rubrique.

Si un ID utilisateur autre que celui enregistré comme propriétaire d'un abonnement tente de reprendre ou de modifier un abonnement *SOFUID* , l'appel échoue avec RC2434 . L'ID utilisateur propriétaire d'un abonnement peut être affiché à l'aide de la commande **DISPLAY SBSTATUS** .

Ne spécifiez pas à la fois *SOAUID* et *SOFUID*. Si aucun n'est spécifié, la valeur par défaut est *SOFUID*.

**Options de publication:** Les options suivantes contrôlent la manière dont les publications sont envoyées à cet abonné. Si vous modifiez un abonnement existant à l'aide de l'option *SOALT* , ces options de publication peuvent être modifiées.

## **SONOLC**

Indique au courtier que l'application ne souhaite pas voir ses propres publications. Les publications sont considérées comme provenant de la même application si les descripteurs de connexion sont identiques. En cas de retour d'un appel *MQSUB* à l'aide de *SORES* , cette option est définie si nécessaire.



## SONEWP

Aucune publication actuellement conservée ne doit être envoyée, lorsque cet abonnement est créé, uniquement les nouvelles publications. Cette option s'applique uniquement lorsque SOCRE est spécifié. Les modifications apportées ultérieurement à un abonnement ne modifient pas le flux des publications et, par conséquent, les publications qui ont été conservées sur une rubrique ont déjà été envoyées à l'abonné en tant que nouvelles publications.

Si cette option est spécifiée sans SOCRE, l'appel échoue avec RC2046. En cas de retour d'un appel MQSUB à l'aide de SORES, cette option n'est pas définie même si l'abonnement a été créé à l'aide de cette option.

Si cette option n'est pas utilisée, les messages précédemment conservés sont envoyés à la file d'attente de destination fournie. Si cette action échoue en raison d'une erreur, RC2525 ou RC2526, la création de l'abonnement échoue.

Cette option n'est pas valide en combinaison avec SOPUBR.

## SOPUBR

La définition de cette option indique que l'abonné demande des informations spécifiquement lorsque cela est nécessaire. Le gestionnaire de files d'attente n'envoie pas de messages non sollicités à l'abonné. La publication conservée (ou éventuellement plusieurs publications si un caractère générique est spécifié dans la rubrique) est envoyée à l'abonné chaque fois qu'un appel MQSUBRQ est effectué à l'aide du descripteur Hsub d'un appel MQSUB précédent. Aucune publication n'est envoyée suite à l'appel MQSUB utilisant cette option. En cas de retour d'un appel MQSUB à l'aide de SORES, cette option est définie si nécessaire.

Cette option n'est pas valide avec SONEWP.

**Options génériques:** Les options suivantes contrôlent la façon dont les caractères génériques sont interprétés dans la chaîne fournie dans la zone SDOS du MQSD. Vous ne pouvez spécifier qu'une seule de ces options. Si vous modifiez un abonnement existant à l'aide de l'option SOALT, ces options génériques ne peuvent pas être modifiées. En cas de retour d'un appel MQSUB à l'aide de SORES, l'option de caractère générique appropriée est définie.

## SOWCHR (RSC)

Les caractères génériques ne s'appliquent qu'aux caractères de la chaîne de rubrique. La zone SOWCHR traite la barre oblique (/) comme un autre caractère sans signification particulière.

Le comportement défini par SOWCHR est illustré dans le tableau suivant:

Caractère spécial	Comportement
*	Caractère générique, zéro ou plusieurs caractères
?	Caractère générique, un caractère
%	Caractère d'échappement permettant d'utiliser les caractères '*', '?' ou '%' dans une chaîne et de ne pas être interprété comme un caractère spécial, par exemple, '% *', '%?' ou '%%'.

Par exemple, la publication sur la rubrique suivante:

```
/level0/level1/level2/level3/level4
```

correspond aux abonnés à l'aide des rubriques suivantes:

```
*  
/*  
/ level0/level1/level2/level3/*  
/ level0/level1/*/level3/level4  
/ level0/level1/le?e12/level3/level4
```

**Remarque :** Cette utilisation de caractères génériques fournit exactement la signification fournie dans IBM MQ V6 et WebSphere MB V6 lors de l'utilisation de messages MQRFH1 formatés pour la publication / abonnement. Il est recommandé de ne pas l'utiliser pour les applications nouvellement écrites et de ne l'utiliser que pour les applications qui étaient précédemment exécutées sur cette version et qui n'ont pas été modifiées pour utiliser le comportement de caractère générique par défaut, comme décrit dans SOWTOP.

## SOWTOP

Les caractères génériques n'agissent que sur les éléments de rubrique dans la chaîne de rubrique. Il s'agit du comportement par défaut si aucun n'est sélectionné.

Le comportement requis par SOWTOP est illustré dans le tableau suivant:

Tableau 730. Interprétation des caractères génériques	
Caractère spécial	Comportement
/	Séparateur de niveau de rubrique
#	Caractère générique: plusieurs niveaux de rubrique
+	Caractère générique: niveau de rubrique unique

### Remarque :

Les caractères '+et'# ne sont pas traités comme des caractères génériques s'ils sont mélangés avec d'autres caractères (y compris eux-mêmes) au sein d'un niveau de rubrique. Dans la chaîne suivante, les caractères '#et'+ sont traités comme des caractères ordinaires.

```
level0/level1/#+/level3/level1#
```

Par exemple, la publication sur la rubrique suivante:

```
/level0/level1/level2/level3/level4
```

correspond aux abonnés à l'aide des rubriques suivantes:

```
#
/#
/ level0/level1/level2/level3/#
/ level0/level1+/level3/level4
```

**Remarque :** Cette utilisation de caractères génériques fournit la signification fournie dans WebSphere Message Broker 6 lors de l'utilisation de messages au format MQRFH2 pour la publication / abonnement.

**Autres options:** Les options suivantes contrôlent la manière dont l'appel d'API est émis et non l'abonnement. En cas de retour d'un appel MQSUB à l'aide de SORES, ces options sont inchangées.

### SOALTU

La zone SDAU contient un identificateur utilisateur à utiliser pour valider cet appel MQSUB. L'appel ne peut aboutir que si cette unité SDAU est autorisée à ouvrir l'objet avec les options d'accès spécifiées, que l'ID utilisateur sous lequel l'application est exécutée soit ou non autorisé à le faire.

### SOSCID

L'abonnement doit utiliser l'identificateur de corrélation fourni dans la zone *SDCID*. Si cette option n'est pas spécifiée, un identificateur de corrélation est automatiquement créé par le gestionnaire de files d'attente au moment de l'abonnement et renvoyé à l'application dans la zone *SDCID*. Pour plus d'informations, voir [SDCID \(chaîne de 24 octets\) SDCID](#).

## SOSETI

L'abonnement doit utiliser le jeton de comptabilité et les données d'identité d'application fournies dans les zones *SDACC* et *SDAID*.

Si cette option est spécifiée, le même contrôle d'autorisation est effectué comme si la file d'attente de destination était accessible à l'aide d'un appel MQOPEN avec 00SETI, sauf dans le cas où l'option SOMAN est également utilisée, auquel cas il n'y a pas de contrôle d'autorisation sur la file d'attente de destination.

Si cette option n'est pas spécifiée, les informations de contexte par défaut sont associées aux publications envoyées à cet abonné comme suit:

Zone dans MQMD	Valeur utilisée
<i>MDUID</i>	ID utilisateur associé à l'abonnement au moment où l'abonnement a été effectué.
<i>MDACC</i>	Déterminé à partir de l'environnement si possible ; défini sur ACNONE si ce n'est pas le cas.
<i>MDAID</i>	Mettre à blanc

Cette option est valide uniquement avec SOCRE et SOALT. Si cette option est utilisée avec SORES, les zones *SDACC* et *SDAID* sont ignorées, de sorte que cette option n'a aucun effet.

Si un abonnement est modifié sans utiliser cette option alors que l'abonnement avait précédemment fourni des informations de contexte d'identité, des informations de contexte par défaut sont générées pour l'abonnement modifié.

Si un abonnement permettant à différents ID utilisateur de l'utiliser avec l'option SOAUIDest repris par un autre ID utilisateur, le contexte d'identité par défaut est généré pour le nouvel ID utilisateur propriétaire de l'abonnement et toutes les publications suivantes sont distribuées contenant le nouveau contexte d'identité.

## SOFIQ

L'appel MQSUB échoue si le gestionnaire de files d'attente est à l'état de mise au repos. Sous z/OS, pour une application CICS ou IMS, cette option force également l'échec de l'appel MQSUB si la connexion est à l'état de mise au repos.

## SDAU (chaîne de caractères de 12 octets)

Si vous spécifiez SOALTU, cette zone contient un autre identificateur utilisateur qui est utilisé pour vérifier l'autorisation de l'abonnement et de la sortie dans la file d'attente de destination (spécifiée dans le paramètre **Hobj** de l'appel MQSUB), à la place de l'identificateur utilisateur sous lequel l'application est en cours d'exécution.

Si l'opération aboutit, l'ID utilisateur indiqué dans cette zone est enregistré en tant qu'ID utilisateur propriétaire de l'abonnement à la place de l'ID utilisateur sous lequel l'application est en cours d'exécution.

Si SOALTU est spécifié et que cette zone est entièrement vide jusqu'au premier caractère null ou jusqu'à la fin de la zone, l'abonnement peut aboutir uniquement si aucune autorisation utilisateur n'est requise pour s'abonner à cette rubrique avec les options spécifiées ou la file d'attente de destination pour la sortie.

Si SOALTU n'est pas spécifié, cette zone est ignorée.

En cas de retour d'un appel MQSUB à l'aide de SORES, cette zone n'est pas modifiée.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par LNUID. La valeur initiale de cette zone est de 12 caractères blancs.

### **SDPRI (entier signé à 10 chiffres)**

Il s'agit de la valeur qui se trouve dans la zone *MQPRI* du descripteur de message (MQMD) de tous les messages de publication correspondant à cet abonnement. Pour plus d'informations sur la zone *MQPRI* dans le MQMD, voir [MDPRI](#).

La valeur doit être supérieure ou égale à zéro ; zéro est la priorité la plus basse. Les valeurs spéciales suivantes peuvent également être utilisées:

#### **PRQDEF**

Lorsqu'une file d'attente d'abonnement est fournie dans la zone *Hobj* de l'appel MQSUB et qu'elle n'est pas un descripteur géré, la priorité du message est extraite de l'attribut **DefPriority** de cette file d'attente. Si la file d'attente ainsi identifiée est une file d'attente de cluster ou qu'il existe plusieurs définitions dans le chemin de résolution de nom de file d'attente, la priorité est déterminée lorsque le message de publication est inséré dans la file d'attente, comme décrit pour [MDPRI](#).

Si l'appel MQSUB utilise un descripteur géré, la priorité du message est extraite de l'attribut **DefPriority** de la file d'attente modèle associée à la rubrique à laquelle il est abonné.

#### **PPUB**

La priorité du message est la priorité de la publication d'origine. Il s'agit de la valeur initiale de la zone.

Si vous modifiez un abonnement existant à l'aide de l'option SOALT , le *MQPRI* des messages de publication ultérieurs peut être modifié.

En cas de retour d'un appel MQSUB à l'aide de SORES, cette zone est définie sur la priorité en cours utilisée pour l'abonnement.

### **SDRO (MQCHARV)**

SDRO est le nom d'objet long après que le gestionnaire de files d'attente a résolu le nom fourni dans *SDON*.

Si le nom d'objet long est fourni dans *SDOS* et que rien n'est fourni dans *SDON*, la valeur renvoyée dans cette zone est la même que celle fournie dans *SDOS*.

Si cette zone est omise (c'est-à-dire *SDRO.VSBufSize* est égal à zéro), *SDRO* n'est pas renvoyé, mais la longueur est renvoyée dans *SDRO.VSLength*. Si la longueur est inférieure à la longueur *SDRO* complète, elle est tronquée et renvoie autant de caractères les plus à droite que la longueur indiquée.

Si *SDRO* est spécifié de manière incorrecte, conformément à la description de l'utilisation de la structure [MQCHARV](#) , ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie RC2520 .

### **SDSID (chaîne de caractères de 4 octets)**

Il s'agit de l'identificateur de structure ; la valeur doit être:

#### **SDSIDV**

Identificateur de la structure du descripteur d'abonnement.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est SDSIDV

### **SDSL (entier signé à 10 chiffres)**

Niveau associé à l'abonnement. Les publications ne sont distribuées à cet abonnement que si elles se trouvent dans l'ensemble d'abonnements dont la valeur *SDSL* la plus élevée est inférieure ou égale à la valeur *PubLevel* utilisée au moment de la publication.

La valeur doit être comprise entre zéro et 9. Zéro est la valeur la plus faible.

La valeur initiale de cette zone est 1.

Si vous modifiez un abonnement existant à l'aide de l'option SOALT , *SDSL* ne peut pas être modifié.

## **SDSN (MQCHARV)**

SDSN indique le nom de l'abonnement.

Cette zone est requise uniquement si *SDOPT* spécifie l'option SODUR , mais si elle est fournie, elle est également utilisée par le gestionnaire de files d'attente pour SONDUR . S'il est spécifié, *SDSN* doit être unique dans le gestionnaire de files d'attente, car il s'agit de la zone utilisée pour identifier les abonnements.

La longueur maximale de *SDSN* est 10240.

Cette zone a deux objectifs. Pour un abonnement SODUR, il s'agit du moyen par lequel vous identifiez un abonnement pour le reprendre après sa création, si vous avez fermé l'indicateur de l'abonnement (à l'aide de l'option COKPSB ) ou si vous avez été déconnecté du gestionnaire de files d'attente. L'identification d'un abonnement à supprimer après sa création est effectuée à l'aide de l'appel MQSUB avec l'option SORES . La zone SDSN est également affichée dans la vue d'administration des abonnements dans la zone *SDSN* de DISPLAY SBSTATUS.

Si *SDSN* est spécifié de manière incorrecte, selon la description du mode d'utilisation de la structure MQCHARV , ou s'il dépasse la longueur maximale, ou s'il est omis lorsqu'il est requis (c'est-à-dire *SDSN*). *VCHRL* est égal à zéro), ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie RC2440 .

Il s'agit d'une zone d'entrée. Les valeurs initiales des zones de cette structure sont identiques à celles de la structure MQCHARV.

Si vous modifiez un abonnement existant à l'aide de l'option SOALT , le nom de l'abonnement ne peut pas être modifié car il s'agit de la zone utilisée pour identifier l'abonnement. Il n'est pas modifié dans la sortie d'un appel MQSUB avec l'option SORES .

## **SDSS (MQCHARV)**

SDSS est la chaîne qui fournit les critères de sélection utilisés lors de l'abonnement aux messages d'une rubrique.

Cette zone de longueur variable est renvoyée en sortie d'un appel MQSUB à l'aide de l'option SORES , si une mémoire tampon est fournie et s'il existe également une longueur de mémoire tampon positive dans VSBufSize. Si aucune mémoire tampon n'est fournie lors de l'appel, seule la longueur de la chaîne de sélection est renvoyée dans la zone VSLength de MQCHARV. Si la mémoire tampon fournie est inférieure à l'espace requis pour renvoyer la zone, seuls les octets VSBufSize sont renvoyés dans la mémoire tampon fournie.

Si *SDSS* est spécifié de manière incorrecte, conformément à la description de l'utilisation de la structure MQCHARV , ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie RC2519 .

## **SDSUD (MQCHARV)**

Les données fournies sur l'abonnement dans cette zone sont incluses en tant que propriété de message mq.SubUserData de chaque publication envoyée à cet abonnement.

La longueur maximale de *SDSUD* est 10240.

Si *SDSUD* est spécifié de manière incorrecte, selon la description de l'utilisation de la structure MQCHARV , ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie RC2431.

Il s'agit d'une zone d'entrée. Les valeurs initiales des zones de cette structure sont identiques à celles de la structure MQCHARV.

Si vous modifiez un abonnement existant à l'aide de l'option SOALT , les données utilisateur de l'abonnement peuvent être modifiées.

Cette zone de longueur variable est renvoyée en sortie d'un appel MQSUB à l'aide de l'option SORES , si une mémoire tampon est fournie et que la longueur de la mémoire tampon est positive dans *VSBufLen*. Si aucune mémoire tampon n'est fournie sur l'appel, seule la longueur des données utilisateur d'abonnement est renvoyée dans la zone *VCHRL* de MQCHARV. Si la mémoire tampon

fournie est inférieure à l'espace requis pour renvoyer la zone, seuls *VSBuflen* octets sont renvoyés dans la mémoire tampon fournie.

### SDVER (entier signé à 10 chiffres)

Il s'agit du numéro de version de la structure ; la valeur doit être:

#### SDVER1

Structure de descripteur d'abonnement Version-1 .

La constante suivante indique le numéro de version de la version en cours:

#### SDVERC

Version actuelle de la structure du descripteur d'abonnement.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de la zone est SDVER1.

## Valeurs initiales

Tableau 732. Zones dans MQSD		
Nom de zone	Nom de la constante	Valeur de la constante
SDSID	SDSIDV	'SD↵↵'
SDVER	SDVER1	1
SDOPT	SONDUR	0
SDON	Aucun	Espaces vides
SDAU	Aucun	Espaces vides
SDASI	SINONE	Valeurs NULL
SDEXP	EIULIM	-1
SDOS	Noms et valeurs définis pour MQCHARV	
SDSN	Noms et valeurs définis pour MQCHARV	
SDSUD	Noms et valeurs définis pour MQCHARV	
SDCID	CINONE	Valeurs NULL
SDPRI	PRQDEF	-3
SDACC	ACNONE	Valeurs NULL
SDAID	Aucun	Espaces vides
SDSL	Aucun	1
SDRO	Noms et valeurs tels que définis dans MQCHARV	
<b>Remarque :</b>		
1. Le symbole ↵ représente un caractère blanc unique.		

## Déclaration RPG

D\*..1.....2.....3.....4.....5.....6.....7..  
D\* MQSD Structure  
D\*

```

D* Structure identifier
D SDSID 1 4
D* Structure version number
D SDVER 5 8I 0
D* Options associated with subscribing
D SDOPT 9 12I 0
D* Object name
D SDON 13 60
D* Alternate user identifier
D SDAU 61 72
D* Alternate security identifier
D SDASI 73 112
D* Expiry of Subscription
D SDEXP 113 116I 0
D* Object Long name
D SDOSP 117 132*
D SDOSO 133 136I 0
D SDOSS 137 140I 0
D SDOSL 141 144I 0
D SDOSC 145 148I 0
D* Subscription name
D SDSNP 149 164*
D SDSNO 165 168I 0
D SDSNS 169 172I 0
D SDSNL 173 176I 0
D SDSNC 177 180I 0
D* Subscription User data
D SDSUDP 181 196*
D SDSUDO 197 200I 0
D SDSUDS 201 204I 0
D SDSUDL 205 208I 0
D SDSUDC 209 212I 0
D* Correlation Id related to this subscription
D SDCID 213 236
D* Priority set in publications
D SDPRI 237 240I 0
D* Accounting Token set in publications
D SDACC 241 272
D* Appl Identity Data set in publications
D SDAID 273 304
D* Message Selector
D SDSSP 305 320*
D SDSSO 321 324I 0
D SDSSS 325 328I 0
D SDSSL 329 332I 0
D SDSSC 333 336
D* Subscription level
D SDSL 337 340 0
D* Resolved Long object name
D SDROP 341 356*
D SDR00 357 360I 0
D SDR0S 361 364I 0
D SDR0L 365 368I 0
D SDR0C 369 372I 0

```

## MQSMPO (définition des options de propriété de message) sous IBM i

La structure **MQSMPO** permet aux applications de spécifier des options qui contrôlent la manière dont les propriétés des messages sont définies.

### Présentation

**Objectif:** la structure est un paramètre d'entrée dans l'appel **MQSETMP** .

**Jeu de caractères et codage:** les données dans **MQSMPO** doivent être dans le jeu de caractères de l'application et le codage de l'application (ENNAT).

- «Zones», à la page [1288](#)
- «Valeurs initiales», à la page [1289](#)
- «Déclaration RPG», à la page [1289](#)

## Zones

La structure MQSMPO contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

### **SPOPT (entier signé à 10 chiffres)**

**Options d'emplacement:** Les options suivantes se rapportent à l'emplacement relatif de la propriété par rapport au curseur de propriété:

#### **SPSETF**

Définit la valeur de la première propriété qui correspond au nom spécifié ou, si elle n'existe pas, ajoute une nouvelle propriété après toutes les autres propriétés avec une hiérarchie correspondante.

#### **SPSETC**

Définit la valeur de la propriété indiquée par le curseur de propriété. La propriété indiquée par le curseur de propriété est celle qui a été interrogée pour la dernière fois à l'aide de l'option IPINQF ou IPINQN.

Le curseur de propriété est réinitialisé lorsque le descripteur de message est réutilisé ou lorsque le descripteur de message est spécifié dans la zone *HMSG* de la structure MQGMO lors d'un appel MQGET ou de la structure MQPMO lors d'un appel MQPUT.

Si cette option est utilisée lorsque le curseur de propriété n'a pas encore été établi ou si la propriété pointée par le curseur de propriété a été supprimée, l'appel échoue avec le code achèvement CCFAIL et le code anomalie RC2471.

#### **SPSETA**

Définit une nouvelle propriété après la propriété pointée par le curseur de propriété. La propriété indiquée par le curseur de propriété est celle qui a été interrogée pour la dernière fois à l'aide de l'option IPINQF ou IPINQO.

Le curseur de propriété est réinitialisé lorsque le descripteur de message est réutilisé ou lorsque le descripteur de message est spécifié dans la zone *HMSG* de la structure MQGMO lors d'un appel MQGET ou de la structure MQPMO lors d'un appel MQPUT.

Si cette option est utilisée lorsque le curseur de propriété n'a pas encore été établi ou si la propriété pointée par le curseur de propriété a été supprimée, l'appel échoue avec le code achèvement CCFAIL et le code anomalie RC2471.

Si vous n'avez besoin d'aucune des options décrites, utilisez l'option suivante:

#### **AUCUN**

Aucune option spécifiée.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est SPSETF.

### **SPSID (entier signé à 10 chiffres)**

Il s'agit de l'identificateur de structure ; la valeur doit être:

#### **SPSIDV**

Identificateur de la structure des options de définition des propriétés de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **SPSIDV**.

### **SPVAKCSI (entier signé à 10 chiffres)**

Jeu de caractères de la valeur de propriété à définir si la valeur est une chaîne de caractères.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **CSAPL**.

### **SPVALENC (entier signé à 10 chiffres)**

Codage de la valeur de propriété à définir si la valeur est numérique.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **ENNAT**.



## SPVER (entier signé à 10 chiffres)

Il s'agit du numéro de version de la structure ; la valeur doit être:

### SPVER1

Version-1 définit la structure des options de propriété de message.

La constante suivante indique le numéro de version de la version en cours:

### Instruction SPVERC

Version actuelle de la structure des options de définition des propriétés de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **SPVER1**.

## Valeurs initiales

Nom de zone	Nom de la constante	Valeur de la constante
SPSID	SPSIDV	'SMPO'
SPVER	SPVER1	1
SPOPT	AUCUN	0
SPVALENC	ENNAT	Dépend de l'environnement
SPVALCSI	CSAPL	-3

## Déclaration RPG

```
D* MQSMPO Structure
D*
D*
D* Structure identifier
D  SP SID      1      4  INZ('SMPO')
D*
D* Structure version number
D  SPVER      5      8I 0 INZ(1)
D*
** Options that control the action of
D* MQSETMP
D  SPOPT      9      12I 0 INZ(0)
D*
D* Encoding of Value
D  SPVALENC  13      16I 0 INZ(273)
D*
D* Character set identifier of Value
D  SPVALCSI  17      20I 0 INZ(-3)
```

## IBM i MQSRO (Subscription Request Options) sur IBM i

La structure MQSRO permet à l'application de spécifier des options qui contrôlent la manière dont une demande d'abonnement est effectuée.

## Présentation

**Objectif:** La structure est un paramètre d'entrée-sortie dans l'appel MQSUBRQ.

**Version:** la version actuelle de MQSRO est SRVER1.

- [«Zones», à la page 1290](#)
- [«Valeurs initiales», à la page 1290](#)
- [«Déclaration RPG», à la page 1291](#)

## Zones

La structure MQSRO contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

### SRNMP (entier signé à 10 chiffres)

Il s'agit d'une zone de sortie renvoyée à l'application pour indiquer le nombre de publications envoyées à la file d'attente d'abonnement à la suite de cet appel. Bien que ce nombre de publications ait été envoyé à la suite de cet appel, rien ne garantit que ce nombre de messages sera disponible pour l'application, en particulier s'il s'agit de messages non persistants.

Il peut y avoir plusieurs publications si la rubrique à laquelle vous êtes abonné contient un caractère générique. Si aucun caractère générique n'était présent dans la chaîne de rubrique lorsque l'abonnement représenté par *HSUB* a été créé, une seule publication au maximum est envoyée suite à cet appel.

### SROPT (entier signé à 10 chiffres)

L'une des options suivantes doit être spécifiée. Vous ne pouvez indiquer qu'une seule option.

**Autres options:** l'option suivante contrôle ce qui se passe lorsque le gestionnaire de files d'attente est mis au repos:

#### File d'attente SRFIQ

L'appel MQSUBRQ échoue si le gestionnaire de files d'attente est à l'état de mise au repos.

**Option par défaut:** Si l'option décrite précédemment n'est pas requise, l'option suivante doit être utilisée:

#### AUCUN

Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. Toutes les options sont définies sur leur valeur par défaut.

SRNONE aide à la documentation du programme. Bien que cette option ne soit pas destinée à être utilisée avec une autre option, car sa valeur est zéro, cette utilisation ne peut pas être détectée.

### SRSID (chaîne de caractères de 4 octets)

Il s'agit de l'identificateur de structure ; la valeur doit être:

#### SRSIDV

Identificateur de la structure SROPT de la demande d'abonnement.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est SRSIDV.

### SRVER (entier signé à 10 chiffres)

Il s'agit du numéro de version de la structure ; la valeur doit être:

#### SRVER1

Structure des options de demande d'abonnement Version-1 .

La constante suivante indique le numéro de version de la version en cours:

#### SRVERC

Version actuelle de la structure des options de demande d'abonnement.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est SRVER1.

## Valeurs initiales

Nom de zone	Nom de la constante	Valeur de la constante
SRSID	SRSIDV	'SRO-'
SRVER	SRVER1	1

Tableau 734. Zones dans MQSRO (suite)

Nom de zone	Nom de la constante	Valeur de la constante
SROPT	AUCUN	0
SRNMP	Aucun	0

**Remarques :**

1. Le symbole `␣` représente un caractère blanc unique.
2. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C, et les caractères blancs dans les autres langages de programmation.

## Déclaration RPG

```

D*..1.....2.....3.....4.....5.....6.....7..
D* MQSRO Structure
D*
D* Structure identifier
D SRSID          1          4
D* Structure version number
D SRVER          5          8I 0
D* Options that control the action of MQSUBRQ
D SROPT          9          12I 0
D* Number of publications sent
D SRNMP         13          16I 0
    
```



## MQSTS (structure de génération de rapports de statut) sous IBM i

La structure MQSTS décrit les données de la structure de statut renvoyée par la commande MQSTAT.

### Présentation

**Jeu de caractères et codage:** les données de type caractère dans MQSTS se trouvent dans le jeu de caractères du gestionnaire de files d'attente local ; cette valeur est fournie par l'attribut de gestionnaire de files d'attente *CodedCharSetId* . Les données numériques dans MQSTS sont dans le codage de la machine native ; elles sont fournies par *ENNAT*.

**Syntaxe:** La commande MQSTAT permet d'extraire des informations de statut. Ces informations sont renvoyées dans une structure MQSTS. Pour plus d'informations sur MQSTAT, voir [«MQSTAT \(Extraire les informations de statut\) sur IBM i»](#), à la page 1425.

- [«Zones»](#), à la page 1291
- [«Valeurs initiales»](#), à la page 1295
- [«Déclaration RPG»](#), à la page 1295

### Zones

La structure MQSTS contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

#### STSCC (entier signé à 10 chiffres)

Il s'agit du code achèvement résultant de la première erreur signalée dans la structure MQSTS.

Il s'agit toujours d'une zone de sortie. La valeur initiale de ce champ est CCOK.

#### STSFC (entier signé à 10 chiffres)

Nombre d'appels d'insertion asynchrones ayant échoué.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0.

### **STSOBJN (chaîne de caractères de 48 octets)**

Il s'agit du nom local de l'objet impliqué dans le premier échec.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est de 48 caractères blancs.

### **STSOQGR (chaîne de caractères de 48 octets)**

Il s'agit du nom du gestionnaire de files d'attente sur lequel l'objet *STSOBJN* est défini. Un nom entièrement vide jusqu'au premier caractère NULL ou jusqu'à la fin de la zone indique le gestionnaire de files d'attente auquel l'application est connectée (le gestionnaire de files d'attente local).

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est de 48 caractères blancs.

### **STS00 (entier signé à 10 chiffres)**

STS00 utilisé pour ouvrir l'objet faisant l'objet d'un rapport. Présent uniquement dans la version 2 de MQSTS ou ultérieure.

La valeur de STS00 dépend de la valeur du paramètre MQSTAT **STYPE** .

#### **STATAPT**

Zéro.

#### **ERRE\_STATUT**

Zéro.

#### **STATUT**

STS00 utilisé lorsque l'incident s'est produit. La raison de l'échec est indiquée dans les zones *STSCC* et *STSRC* de la structure MQSTS .

STS00 est un champ de sortie. Sa valeur initiale est zéro.

### **STSOS (MQCHARV)**

Nom d'objet long de l'objet défaillant signalé. Présent uniquement dans la version 2 de MQSTS ou ultérieure.

STSOS est une zone MQCHARV d'une longueur maximale de 10240 caractères. Voir [MQCHARV](#) pour une description de l'utilisation de la structure MQCHARV.

L'interprétation de STSOS dépend de la valeur du paramètre MQSTAT **STYPE** .

#### **STATAPT**

Il s'agit du nom d'objet long de la file d'attente ou de la rubrique utilisée dans l'opération MQPUT , qui a échoué.

#### **ERRE\_STATUT**

Chaîne de longueur zéro

#### **STATUT**

Il s'agit du nom d'objet long de l'objet qui a provoqué l'échec de la reconnexion.

STSOS est un champ de sortie. Sa valeur initiale est une chaîne de longueur nulle.

### **STSOT (entier signé à 10 chiffres)**

Type d'objet nommé dans *ObjectName*. Les valeurs possibles sont les suivantes :

#### **OTALSQ**

File d'attente alias.

#### **OTLOCQ**

File d'attente locale.

#### **OTMODQ**

File d'attente modèle.

**File d'attente OTQ**

File d'attente.

**OTREMQ**

File d'attente éloignée.

**Mot de passe OTTOP**

:NONE.

Il s'agit toujours d'une zone de sortie. La valeur initiale de cette zone est OTQ.

**STSRC (entier signé à 10 chiffres)**

Il s'agit du code anomalie résultant de la première erreur signalée dans la structure MQSTS

Il s'agit toujours d'une zone de sortie. La valeur initiale de cette zone est RCNONE.

**STSRBJN (chaîne de caractères de 48 octets)**

Il s'agit du nom de la file d'attente de destination nommée dans *STSRBJN* une fois que le gestionnaire de files d'attente local a résolu le nom. Le nom renvoyé est le nom d'une file d'attente qui existe sur le gestionnaire de files d'attente identifié par *STSRQMGR*.

Une valeur non vide est renvoyée uniquement si l'objet est une file d'attente unique ouverte pour l'exploration, l'entrée ou la sortie (ou toute combinaison). Si l'objet ouvert est l'un des suivants, *STSRBJN* est mis à blanc:

- Une rubrique
- Une file d'attente, mais non ouverte pour l'exploration, l'entrée ou la sortie

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est de 48 caractères blancs.

**STSRQMGR (chaîne de caractères de 48 octets)**

Il s'agit du nom du gestionnaire de files d'attente de destination une fois que le gestionnaire de files d'attente local a résolu le nom. Le nom renvoyé est le nom du gestionnaire de files d'attente qui possède la file d'attente identifiée par *STSRBJN*. *STSRQMGR* peut être le nom du gestionnaire de files d'attente local.

Si *STSRBJN* est une file d'attente partagée appartenant au groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local, *STSRQMGR* est le nom du groupe de partage de files d'attente. Si la file d'attente appartient à un autre groupe de partage de files d'attente, *STSRBJN* peut être le nom du groupe de partage de files d'attente ou le nom d'un gestionnaire de files d'attente membre du groupe de partage de files d'attente (la nature de la valeur renvoyée est déterminée par les définitions de files d'attente qui existent sur le gestionnaire de files d'attente local).

Une valeur non vide est renvoyée uniquement si l'objet est une file d'attente unique ouverte pour l'exploration, l'entrée ou la sortie (ou toute combinaison). Si l'objet ouvert est l'un des suivants, *STSRQMGR* est mis à blanc:

- Une rubrique
- Une file d'attente, mais non ouverte pour l'exploration, l'entrée ou la sortie
- Une file d'attente de cluster avec OOBNDN spécifié (ou avec OOBNDQ en vigueur lorsque l'attribut de file d'attente **DefBind** a la valeur OOBNDN)

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est de 48 caractères blancs.

**STSSC (entier signé à 10 chiffres)**

Nombre d'appels d'insertion asynchrones ayant abouti.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0.

**STSSID (chaîne de caractères de 4 octets)**

Il s'agit de l'identificateur de structure. La valeur doit être:

**STSSID**

Identificateur de la structure de génération de rapports de statut.

La valeur initiale de cette zone est STSSID.

**STSSO (entier signé à 10 chiffres)**

STSSO utilisé pour ouvrir l'abonnement défaillant. Présent uniquement dans la version 2 de MQSTS ou ultérieure.

L'interprétation de STSSO dépend de la valeur du paramètre MQSTAT **STYPE** .

**STATAPT**

Zéro.

**ERRE\_STATUT**

Zéro.

**STATUT**

STSSO utilisé lorsque l'incident s'est produit. La raison de l'échec est indiquée dans les zones *STSCC* et *STSRC* de la structure MQSTS . Si l'échec n'est pas lié à l'abonnement à une rubrique, la valeur renvoyée est zéro.

STSSO est un champ de sortie. Sa valeur initiale est zéro.

**STSSUN (MQCHARV)**

Nom de l'abonnement défaillant. Présent uniquement dans la version 2 de MQSTS ou ultérieure.

STSSUN est une zone MQCHARV d'une longueur maximale de 10240 caractères. Voir [MQCHARV](#) pour une description de l'utilisation de la structure MQCHARV.

L'interprétation de STSSUN dépend de la valeur du paramètre MQSTAT **STYPE** .

**STATAPT**

Chaîne de longueur zéro.

**ERRE\_STATUT**

Chaîne de longueur zéro.

**STATUT**

Nom de l'abonnement qui a provoqué l'échec de la reconnexion. Si aucun nom d'abonnement n'est disponible ou si l'échec n'est pas lié à un abonnement, il s'agit d'une chaîne de longueur nulle.

STSSUN est un champ de sortie. Sa valeur initiale est une chaîne de longueur nulle.

**STSVR (entier signé à 10 chiffres)**

Il s'agit du numéro de version de la structure. La valeur doit être:

**STSVR1**

Numéro de version de la structure de génération de rapports de statut.

La constante suivante indique le numéro de version de la version en cours:

**Commande STSVRC**

Version actuelle de la structure de génération de rapports de statut.

La valeur initiale de cette zone est STSVR1.

**STCFC (entier signé à 10 chiffres)**

Nombre d'appels d'insertion asynchrones ayant abouti avec un avertissement.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0.

## Valeurs initiales

Tableau 735. Zones dans MQSTS		
Nom de zone	Nom de la constante	Valeur de la constante
STSSID	ID de sous-système	
STSVR	Commande STSVRC	STSVR1
STSCC	CCOK	0
STSRC	RCNONE	0
STSSC	Aucun	0
STSWC	Aucun	0
STSFC	Aucun	0
STSOT	Aucun	0
STSOBJN	Aucun	Espaces vides
STSOQMGR	Aucun	Espaces vides
STSR OBJN	Aucun	Espaces vides
STSRQMGR	Aucun	Espaces vides
STSOS	Noms et valeurs définis pour MQCHARV	
STSSUN	Noms et valeurs définis pour MQCHARV	
STS00	Aucun	0
STSS0	Aucun	0

## Déclaration RPG

```

D*..1.....2.....3.....4.....5.....6.....7..
D* MQSTS Structure
D*
D* Structure identifier
D STSSID          1      4
D* Structure version number
D STSVR           5      8I 0
D* Completion code
D STSCC           9      12I 0
D* Reason code
D STSRC           13     16I 0
D* Success count
D STSSC           17     20I 0
D* Warning count
D STSWC           21     24I 0
D* Failure count
D STSFC           25     28I 0
D* Object type
D STSOT           29     32I 0
D* Object name
D STSOBJN         33     80
D* Object queue manager
D STSOQMGR        81     128
D* Resolved object name
D STSR OBJN       129    176
D* Resolved object queue manager name
D STSRQMGR        177    224
D* Ver:1 **
D* Failing object long name
D* Address of variable length string
D STSOSCHRP       225    240*
D* Offset of variable length string
D STSOSCHRO       241    244I 0

```

```

D* Size of buffer
D STSOSVSBS          245    248I 0
D* Length of variable length string
D STSOSCHRL         249    252I 0
D* CCSID of variable length string
D STSOSCHRC         253    256I 0
D* Failing subscription name
D* Address of variable length string
D STSSUNCHRP        257    272*
D* Offset of variable length string
D STSSUNCHRO        273    276I 0
D* Size of buffer
D STSSUNVSBS        277    280I 0
D* Length of variable length string
D STSSUNCHRL        281    284I 0
D* CCSID of variable length string
D STSSUNCHRC        285    288I 0
D* Failing open options
D STS00             289    292I 0
D* Failing subscription options
D STSS0             293    296I 0
D* Ver:2 **

```

## MQTM-Message de déclenchement

La structure MQTM décrit les données du message de déclenchement envoyé par le gestionnaire de files d'attente à une application de moniteur de déclenchement lorsqu'un événement de déclenchement se produit pour une file d'attente.

### Présentation

**Objectif:** Cette structure fait partie de l'interface TMI ( IBM MQ Trigger Monitor Interface), qui est l'une des interfaces de l'infrastructure IBM MQ .

**Nom de format:** FMTM.

**Jeu de caractères et codage:** les données de type caractère dans MQTM se trouvent dans le jeu de caractères du gestionnaire de files d'attente qui génère le MQTM. Les données numériques dans MQTM se trouvent dans le codage machine du gestionnaire de files d'attente qui génère le MQTM.

Le jeu de caractères et le codage du MQTM sont fournis par les zones *MDCSI* et *MDENC* dans:

- MQMD (si la structure MQTM est au début des données de message), ou
- Structure d'en-tête qui précède la structure MQTM (tous les autres cas).

**Utilisation:** Une application de moniteur de déclenchement peut avoir besoin de transmettre une partie ou la totalité des informations du message de déclenchement à l'application démarrée par l'application de moniteur de déclenchement. Les informations qui peuvent être requises par l'application démarrée incluent *TMQN*, *TMTDet* *TMUD*. L'application de moniteur de déclenchement peut transmettre la structure MQTM directement à l'application démarrée ou transmettre une structure MQTMC2 à la place, en fonction de ce qui est autorisé par l'environnement et pratique pour l'application démarrée. Pour plus d'informations sur MQTMC2, voir [«MQTMC2 \(format de 2 caractères du message de déclenchement\) sous IBM i»](#), à la page 1300.

- Sous IBM i, l'application de moniteur de déclenchement fournie avec IBM MQ transmet une structure MQTMC2 à l'application démarrée.

Pour plus d'informations sur les déclencheurs, voir [Prérequis pour le déclenchement](#).

- [«MQMD pour un message de déclenchement»](#), à la page 1297
- [«Zones»](#), à la page 1297
- [«Valeurs initiales»](#), à la page 1300
- [«Déclaration RPG»](#), à la page 1300



## MQMD pour un message de déclenchement

Tableau 736. Paramètres des zones dans le MQMD d'un message de déclenchement généré par le gestionnaire de files d'attente

Zone dans MQMD	Valeur utilisée
MDSID	MSIDV (MDSID)
MDVER	MDVER1
MDREP	RONONE
MDMT	MTDGRM
MDEXP	EIULIM
MDFB	FBNONE
MDENC	ENNAT
MDCSI	Attribut <b>CodedCharSetId</b> du gestionnaire de files d'attente
MDFMT	FMTM
MDPRI	Attribut <b>DefPriority</b> de la file d'attente d'initialisation
MDPER	PENPER
MDMID	Une valeur unique
MDCID	CINONE
MDBOC	0
MDRQ	Espaces vides
MDRM	Nom du gestionnaire de files d'attente
MDUID	Espaces vides
MDACC	ACNONE
MDAID	Espaces vides
MDPAT	ATQM, ou selon le cas, pour l'agent MCA
MDPAN	28 premiers octets du nom du gestionnaire de files d'attente
MDPD	Date d'envoi du message de déclenchement
MDPT	Heure d'envoi du message de déclenchement
MDAOD	Espaces vides

Une application qui génère un message de déclenchement est recommandée pour définir des valeurs similaires, à l'exception des suivantes:

- La zone *MDPRI* peut être définie sur PRQDEF (le gestionnaire de files d'attente la remplace par la priorité par défaut de la file d'attente d'initialisation lors de l'insertion du message).
- La zone *MDRM* peut être mise à blanc (le gestionnaire de files d'attente change le nom du gestionnaire de files d'attente local lors de l'insertion du message).
- Les zones de contexte doivent être définies en fonction de l'application.

### Zones

La structure MQTM contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

#### TMAI (chaîne de caractères de 256 octets)

Identificateur d'application.

Il s'agit d'une chaîne de caractères qui identifie l'application à démarrer et qui est utilisée par l'application trigger-monitor qui reçoit le message de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut **App1Id** de l'objet de processus identifié par la zone *TMPN* ; pour plus de détails sur cet attribut, voir «Attributs des définitions de processus sous IBM i», à la page 1467 . Le contenu de ces données n'a aucune importance pour le gestionnaire de files d'attente.

La signification de *TMAI* est déterminée par l'application trigger-monitor. Le moniteur de déclenchement fourni par IBM MQ requiert que *TMAI* soit le nom d'un programme exécutable.

La longueur de cette zone est indiquée par LNPROA. La valeur initiale de cette zone est 256 caractères blancs.

### **TMAT (entier signé à 10 chiffres)**

Type d'application.

Identifie la nature du programme à démarrer et est utilisé par l'application du moniteur de déclenchement qui reçoit le message de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut **App1Type** de l'objet de processus identifié par la zone *TMPN* ; pour plus de détails sur cet attribut, voir «Attributs des définitions de processus sous IBM i», à la page 1467 . Le contenu de ces données n'a aucune importance pour le gestionnaire de files d'attente.

*TMAT* peut avoir l'une des valeurs standard suivantes. Les types définis par l'utilisateur peuvent également être utilisés, mais ils doivent être limités aux valeurs comprises entre ATUFST et ATULST:

#### **àCICS**

Transaction CICS .

#### **ATVSE**

Transaction CICS/VSE .

#### **AT400**

Application IBM i .

#### **TATUFST**

Valeur la plus faible pour le type d'application défini par l'utilisateur.

#### **ATULST**

Valeur la plus élevée pour le type d'application défini par l'utilisateur.

La valeur initiale de cette zone est 0.

### **TMED (chaîne de caractères de 128 octets)**

Données d'environnement.

Il s'agit d'une chaîne de caractères qui contient des informations liées à l'environnement concernant l'application à démarrer et qui est utilisée par l'application du moniteur de déclenchement qui reçoit le message de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut **EnvData** de l'objet de processus identifié par la zone *TMPN* ; pour plus de détails sur cet attribut, voir «Attributs des définitions de processus sous IBM i», à la page 1467 . Le contenu de ces données n'a aucune importance pour le gestionnaire de files d'attente.

La longueur de cette zone est indiquée par LNPROE. La valeur initiale de cette zone est de 128 caractères blancs.

### **TMPN (chaîne de caractères de 48 octets)**

Nom de l'objet de processus.

Il s'agit du nom de l'objet de processus de gestionnaire de files d'attente spécifié pour la file d'attente de déclenchement. Il peut être utilisé par l'application du moniteur de déclenchement qui reçoit le message de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut **ProcessName** de la file d'attente identifiée par la zone *TMQN* ; pour plus de détails sur cet attribut, voir «Attributs des files d'attente», à la page 1435 .

Les noms inférieurs à la longueur définie de la zone sont toujours complétés à droite par des blancs ; ils ne sont pas terminés prématurément par un caractère NULL.

La longueur de cette zone est indiquée par LNPRON. La valeur initiale de cette zone est de 48 caractères blancs.

#### **TMQN (chaîne de caractères de 48 octets)**

Nom de la file d'attente déclenchée.

Il s'agit du nom de la file d'attente pour laquelle un événement déclencheur s'est produit. Il est utilisé par l'application démarrée par l'application du moniteur de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut **QName** de la file d'attente de déclenchement ; pour plus de détails sur cet attribut, voir [«Attributs des files d'attente»](#), à la page 1435 .

Les noms inférieurs à la longueur définie de la zone sont complétés à droite par des blancs ; ils ne sont pas terminés prématurément par un caractère NULL.

La longueur de cette zone est indiquée par LNQN. La valeur initiale de cette zone est de 48 caractères blancs.

#### **TMSID (chaîne de caractères de 4 octets)**

Identificateur de structure.

La valeur doit être:

##### **TTMSIDV**

Identificateur de la structure de message de déclenchement.

La valeur initiale de cette zone est TMSIDV.

#### **TMTD (chaîne de caractères de 64 octets)**

Données de déclenchement.

Il s'agit de données au format libre à utiliser par l'application du moniteur de déclenchement qui reçoit le message de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut **TriggerData** de la file d'attente identifiée par la zone *TMQN* ; pour plus de détails sur cet attribut, voir [«Attributs des files d'attente»](#), à la page 1435 . Le contenu de ces données n'a aucune importance pour le gestionnaire de files d'attente.

La longueur de cette zone est indiquée par LNTRGD. La valeur initiale de cette zone est de 64 caractères blancs.

#### **TMUD (chaîne de caractères de 128 octets)**

Données utilisateur.

Il s'agit d'une chaîne de caractères qui contient des informations utilisateur relatives à l'application à démarrer et qui est utilisée par l'application du moniteur de déclenchement qui reçoit le message de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut **UserData** de l'objet de processus identifié par la zone *TMPN* ; pour plus de détails sur cet attribut, voir [«Attributs des définitions de processus sous IBM i»](#), à la page 1467 . Le contenu de ces données n'a aucune importance pour le gestionnaire de files d'attente.

La longueur de cette zone est indiquée par LNPROU. La valeur initiale de cette zone est de 128 caractères blancs.

#### **TMVER (entier signé à 10 chiffres)**

Numéro de version de la structure.

La valeur doit être:

##### **TMVER1**

Numéro de version de la structure de message de déclenchement.

La constante suivante indique le numéro de version de la version en cours:

## TTMVERC

Version actuelle de la structure de message de déclenchement.

La valeur initiale de cette zone est TMVER1.

## Valeurs initiales

Nom de zone	Nom de la constante	Valeur de la constante
TMSID	TTMSIDV	' TM↵ '
TMVER	TMVER1	1
TMQN	Aucun	Espaces vides
TMPN	Aucun	Espaces vides
TMTD	Aucun	Espaces vides
TMAT	Aucun	0
TMAI	Aucun	Espaces vides
TMED	Aucun	Espaces vides
TMUD	Aucun	Espaces vides

**Remarques :**

1. Le symbole ↵ représente un caractère blanc unique.

## Déclaration RPG

```
D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQTM Structure
D*
D* Structure identifier
D TMSID 1 4 INZ(' TM ')
D* Structure version number
D TMVER 5 8I 0 INZ(1)
D* Name of triggered queue
D TMQN 9 56 INZ
D* Name of process object
D TMPN 57 104 INZ
D* Trigger data
D TMTD 105 168 INZ
D* Application type
D TMAT 169 172I 0 INZ(0)
D* Application identifier
D TMAI 173 428 INZ
D* Environment data
D TMED 429 556 INZ
D* User data
D TMUD 557 684 INZ
```

## MQTMC2 (format de 2 caractères du message de déclenchement) sous IBM i

Lorsqu'une application de moniteur de déclenchement extrait un message de déclenchement (MQTM) d'une file d'attente d'initialisation, le moniteur de déclenchement peut avoir besoin de transmettre une partie ou la totalité des informations du message de déclenchement à l'application démarrée par le moniteur de déclenchement.

## Présentation

**Objectif:** les informations qui peuvent être requises par l'application démarrée incluent *TC2QN*, *TC2TDe* et *TC2UD*. L'application du moniteur de déclenchement peut transmettre la structure MQTM directement à l'application démarrée ou transmettre une structure MQTMC2 à la place, en fonction de ce qui est autorisé par l'environnement et pratique pour l'application démarrée.

Cette structure fait partie de l'interface TMI ( IBM MQ Trigger Monitor Interface), qui est l'une des interfaces de l'infrastructure IBM MQ .

**Jeu de caractères et codage:** les données de type caractère dans MQTMC2 se trouvent dans le jeu de caractères du gestionnaire de files d'attente local ; elles sont fournies par l'attribut de gestionnaire de files d'attente **CodedCharSetId** .

**Syntaxe:** la structure MQTMC2 est similaire au format de la structure MQTM. La différence réside dans le fait que les zones non alphanumériques de MQTM sont remplacées dans MQTMC2 par des zones alphanumériques de même longueur et que le nom du gestionnaire de files d'attente est ajouté à la fin de la structure.

- Sous IBM i, l'application du moniteur de déclenchement fournie avec IBM MQ transmet une structure MQTMC2 à l'application démarrée.
- «Zones», à la page [1301](#)
- «Valeurs initiales», à la page [1302](#)
- «Déclaration RPG», à la page [1302](#)

## Zones

La structure MQTMC2 contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

### **TC2AI (chaîne de caractères de 256 octets)**

Identificateur d'application.

Voir la zone *TMAI* dans la structure MQTM.

### **TC2AT (chaîne de caractères de 4 octets)**

Type d'application.

Cette zone contient toujours des blancs, quelle que soit la valeur de la zone *TMAT* dans la structure MQTM du message de déclenchement d'origine.

### **TC2ED (chaîne de caractères de 128 octets)**

Données d'environnement.

Voir la zone *TMED* dans la structure MQTM.

### **TC2PN (chaîne de caractères de 48 octets)**

Nom de l'objet de processus.

Voir la zone *TMPN* dans la structure MQTM.

### **TC2QMN (chaîne de caractères de 48 octets)**

Nom du gestionnaire de files d'attente

Il s'agit du nom du gestionnaire de files d'attente dans lequel l'événement déclencheur s'est produit.

### **TC2QN (chaîne de caractères de 48 octets)**

Nom de la file d'attente déclenchée.

Voir la zone *TMQN* dans la structure MQTM.

### **TC2SID (chaîne de caractères de 4 octets)**

Identificateur de structure.

La valeur doit être:

**TCSIDV**

Identificateur de la structure du message de déclenchement (format de caractères).

**TC2TD (chaîne de caractères de 64 octets)**

Données de déclenchement.

Voir la zone *TMTD* dans la structure MQTM.

**TC2UD (chaîne de caractères de 128 octets)**

Données utilisateur.

Voir la zone *TMUD* dans la structure MQTM.

**TC2VER (chaîne de caractères de 4 octets)**

Numéro de version de la structure.

La valeur doit être:

**TCVER2**

Structure du message de déclenchement version 2 (format de caractère).

La constante suivante indique le numéro de version de la version en cours:

**CTCVERC**

Version en cours de la structure du message de déclenchement (format de caractères).

**Valeurs initiales**

Tableau 738. Zones dans MQTMC2		
Nom de zone	Nom de la constante	Valeur de la constante
TC2SID	TCSIDV	'TMC¬'
TC2VER	TCVER2	'¬¬¬2'
TC2QN	Aucun	Espaces vides
TC2PN	Aucun	Espaces vides
TC2TD	Aucun	Espaces vides
TC2AT	Aucun	Espaces vides
TC2AI	Aucun	Espaces vides
TC2ED	Aucun	Espaces vides
TC2UD	Aucun	Espaces vides
TC2QMN	Aucun	Espaces vides

**Remarques :**

1. Le symbole ¬ représente un caractère blanc unique.

**Déclaration RPG**

```
D*..1.....2.....3.....4.....5.....6.....7..
D* MQTMC2 Structure
D*
D* Structure identifier
D TC2SID          1      4
D* Structure version number
D TC2VER          5      8
```

D*	Name of triggered queue		
D	TC2QN	9	56
D*	Name of process object		
D	TC2PN	57	104
D*	Trigger data		
D	TC2TD	105	168
D*	Application type		
D	TC2AT	169	172
D*	Application identifier		
D	TC2AI	173	428
D*	Environment data		
D	TC2ED	429	556
D*	User data		
D	TC2UD	557	684
D*	Queue manager name		
D	TC2QMN	685	732

IBM i

## MQWIH (en-tête d'informations de travail) sur IBM i

La structure MQWIH décrit les informations qui doivent être présentes au début d'un message qui doit être géré par le gestionnaire de charge de travail z/OS .

### Présentation

**Nom de format:** FMWIH.

**Jeu de caractères et codage:** les zones de la structure MQWIH se trouvent dans le jeu de caractères et le codage donnés par les zones *MDCSI* et *MDENC* de la structure d'en-tête qui précède MQWIH, ou par les zones de la structure MQMD si MQWIH se trouve au début des données de message d'application.

Le jeu de caractères doit comporter des caractères mono-octet pour les caractères admis dans les noms de file d'attente.

**Syntaxe:** si un message doit être traité par le gestionnaire de charge de travail z/OS , il doit commencer par une structure MQWIH.

- «Zones», à la page [1303](#)
- «Valeurs initiales», à la page [1305](#)
- «Déclaration RPG», à la page [1306](#)

### Zones

La structure MQWIH contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

#### WICSI (entier signé à 10 chiffres)

Identificateur de jeu de caractères des données qui suivent MQWIH.

Indique l'identificateur de jeu de caractères des données qui suivent la structure MQWIH ; il ne s'applique pas aux données de type caractères de la structure MQWIH elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. La valeur spéciale suivante peut être utilisée:

#### CSINHT

Hérite de l'identificateur de jeu de caractères de cette structure.

Les données de type caractères dans les données *suivant* cette structure se trouvent dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, la valeur CSINHT n'est pas renvoyée par l'appel MQGET.

CSINHT ne peut pas être utilisé si la valeur de la zone *MDPAT* dans MQMD est ATBRKR.

La valeur initiale de cette zone est CSUNDF.

**WIENC (entier signé à 10 chiffres)**

Codage numérique des données qui suit MQWIH.

Indique le codage numérique des données qui suivent la structure MQWIH ; il ne s'applique pas aux données numériques de la structure MQWIH elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données.

La valeur initiale de cette zone est 0.

**WIFLG (entier signé à 10 chiffres)**

Indicateurs

La valeur doit être:

**WINONE**

Aucun indicateur.

La valeur initiale de cette zone est WINONE.

**WIFMT (chaîne de caractères de 8 octets)**

Nom de format des données qui suivent MQWIH.

Indique le nom de format des données qui suivent la structure MQWIH.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de la zone *MDFMT* dans MQMD.

La longueur de cette zone est indiquée par LNFMT. La valeur initiale de cette zone est FMNONE.

**WILEN (entier signé à 10 chiffres)**

Longueur de la structure MQWIH.

La valeur doit être:

**WILEN1**

Longueur de la structure d'en-tête des informations de travail version-1 .

La constante suivante indique la longueur de la version en cours:

**WILENC**

Longueur de la version actuelle de la structure d'en-tête des informations de travail.

La valeur initiale de cette zone est WILEN1.

**WIRSV (chaîne de caractères de 32 octets)**

Réservé.

Il s'agit d'une zone réservée ; elle doit être vide.

**WISID (chaîne de caractères de 4 octets)**

Identificateur de structure.

La valeur doit être:

**WISIDV**

Identificateur de la structure d'en-tête des informations de travail.

La valeur initiale de cette zone est WISIDV.

**WISNM (chaîne de caractères de 32 octets)**

Nom du service.

Il s'agit du nom du service qui doit traiter le message.



La longueur de cette zone est indiquée par LNSVNM. La valeur initiale de cette zone est de 32 caractères blancs.

### **WISST (chaîne de caractères de 8 octets)**

Nom de l'étape de service.

Il s'agit du nom de l'étape de *WISNM* à laquelle le message est lié.

La longueur de cette zone est indiquée par LNSVST. La valeur initiale de cette zone est de 8 caractères blancs.

### **WITOK (chaîne de 16 octets)**

Jeton de message.

Il s'agit d'un jeton de message qui identifie le message de manière unique.

Pour les appels MQPUT et MQPUT1, cette zone est ignorée. La longueur de cette zone est indiquée par LNMTOK. La valeur initiale de cette zone est MTKNON.

### **WIVER (entier signé à 10 chiffres)**

Numéro de version de la structure.

La valeur doit être:

#### **WIVER1**

Structure d'en-tête des informations de travail Version-1.

La constante suivante indique le numéro de version de la version en cours:

#### **IversC**

Version actuelle de la structure d'en-tête des informations de travail.

La valeur initiale de cette zone est WIVER1.

## **Valeurs initiales**

<i>Tableau 739. Zones dans MQWIH</i>		
<b>Nom de zone</b>	<b>Nom de la constante</b>	<b>Valeur de la constante</b>
<i>WISID</i>	WISIDV	'WIH↵'
<i>WIVER</i>	WIVER1	1
<i>WILEN</i>	WILEN1	120
<i>WIENC</i>	Aucun	0
<i>WICSI</i>	CSUNDF	0
<i>WIFMT</i>	FMAUCUN	Espaces vides
<i>WIFLG</i>	WINONE	0
<i>WISNM</i>	Aucun	Espaces vides
<i>WISST</i>	Aucun	Espaces vides
<i>WITOK</i>	MTKNON	Valeurs NULL
<i>WIRSV</i>	Aucun	Espaces vides
<b>Remarques :</b>		
1. Le symbole ↵ représente un caractère blanc unique.		

## Déclaration RPG

```
D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQWIH Structure
D*
D* Structure identifier
D WISID          1          4      INZ('WIH ')
D* Structure version number
D WIVER          5          8I 0  INZ(1)
D* Length of MQWIH structure
D WILEN          9          12I 0 INZ(120)
D* Numeric encoding of data that followsMQWIH
D WIENC          13         16I 0 INZ(0)
D* Character-set identifier of data thatfollows MQWIH
D WICSI          17         20I 0 INZ(0)
D* Format name of data that followsMQWIH
D WIFMT          21         28      INZ('      ')
D* Flags
D WIFLG          29         32I 0 INZ(0)
D* Service name
D WISNM          33         64      INZ
D* Service step name
D WISST          65         72      INZ
D* Message token
D WITOK          73         88      INZ(X'00000000000000-
D                                     0000000000000000')
D* Reserved
D WIRSV          89         120     INZ
```



## MQXQH (en-tête de file d'attente de transmission) sous IBM i

La structure MQXQH décrit les informations qui sont préfixées aux données de message d'application des messages lorsqu'ils se trouvent dans des files d'attente de transmission.

### Présentation

**Objectif:** Une file d'attente de transmission est un type spécial de file d'attente locale qui contient temporairement des messages destinés à des files d'attente éloignées (c'est-à-dire destinées à des files d'attente qui n'appartiennent pas au gestionnaire de files d'attente local). Une file d'attente de transmission est désignée par l'attribut de file d'attente **Usage** ayant la valeur USTRAN.

**Nom de format:** FMXQH.

**Jeu de caractères et codage:** les données dans MQXQH doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par ENNAT pour le langage de programmation C.

Le jeu de caractères et le codage du MQXQH doivent être définis dans les zones *MDCSI* et *MDENC* dans:

- Le MQMD distinct (si la structure MQXQH est au début des données de message), ou
- Structure d'en-tête qui précède la structure MQXQH (tous les autres cas).

**Utilisation:** Un message qui se trouve dans une file d'attente de transmission comporte *deux* descripteurs de message:

- Un descripteur de message est stocké séparément des données de message ; il est appelé *descripteur de message distinct* et est généré par le gestionnaire de files d'attente lorsque le message est placé dans la file d'attente de transmission. Certaines zones du descripteur de message distinct sont copiées à partir du descripteur de message fourni par l'application sur l'appel MQPUT ou MQPUT1.

Le descripteur de message distinct est celui qui est renvoyé à l'application dans le paramètre **MSGDSC** de l'appel MQGET lorsque le message est supprimé de la file d'attente de transmission.

- Un deuxième descripteur de message est stocké dans la structure MQXQH dans le cadre des données de message ; il s'agit du *descripteur de message intégré* et il s'agit d'une copie du descripteur de message fourni par l'application sur l'appel MQPUT ou MQPUT1 (avec des variations mineures).

Le descripteur de message imbriqué est toujours un MQMD version-1 . Si le message inséré par l'application comporte des valeurs autres que celles par défaut pour une ou plusieurs des zones version-2 du MQMD, une structure MQMDE suit le MQXQH et est à son tour suivie des données de message d'application (le cas échéant). Le MQMDE est l'un des suivants:

- Généré par le gestionnaire de files d'attente (si l'application utilise un MQMD version-2 pour insérer le message), ou
- Déjà présent au début des données de message d'application (si l'application utilise un MQMD version-1 pour insérer le message).

Le descripteur de message imbriqué est celui qui est renvoyé à l'application dans le paramètre **MSGDSC** de l'appel MQGET lorsque le message est supprimé de la file d'attente de destination finale.

- [«Zones du descripteur de message distinct», à la page 1307](#)
- [«Zones du descripteur de message imbriqué», à la page 1308](#)
- [«Insertion de messages dans des files d'attente éloignées», à la page 1309](#)
- [«Insertion de messages directement dans les files d'attente de transmission», à la page 1309](#)
- [«Obtention de messages à partir de files d'attente de transmission», à la page 1309](#)
- [«Zones», à la page 1309](#)
- [«Valeurs initiales», à la page 1310](#)
- [«Déclaration RPG», à la page 1311](#)

## Zones du descripteur de message distinct

Les zones du descripteur de message distinct sont définies par le gestionnaire de files d'attente, comme indiqué dans la liste suivante. Si le gestionnaire de files d'attente ne prend pas en charge le MQMD version-2 , un MQMD version-1 est utilisé sans perte de fonction.

Tableau 740. Zones du descripteur de message distinct et valeurs utilisées

Zone dans MQMD distinct	Valeur utilisée
MDSID	MSIDV (MDSID)
MDVER	MDVER2
MDREP	Copié à partir du descripteur de message imbriqué, mais avec les bits identifiés par ROAUXM définis sur zéro. (Cela empêche la génération d'un message de rapport COA ou COD lorsqu'un message est placé dans ou supprimé d'une file d'attente de transmission.)
MDMT	Copié à partir du descripteur de message imbriqué.
MDEXP	Copié à partir du descripteur de message imbriqué.
MDFB	Copié à partir du descripteur de message imbriqué.
MDENC	ENNAT
MDCSI	Attribut <b>CodedCharSetId</b> du gestionnaire de files d'attente.
MDFMT	FMXQH
MDPRI	Copié à partir du descripteur de message imbriqué.
MDPER	Copié à partir du descripteur de message imbriqué.
MDMID	Une nouvelle valeur est générée par le gestionnaire de files d'attente. Cet identificateur de message est différent du <i>MDMID</i> que le gestionnaire de files d'attente a peut-être généré pour le descripteur de message imbriqué (voir précédemment).
MDCID	<i>MDMID</i> à partir du descripteur de message imbriqué.

Tableau 740. Zones du descripteur de message distinct et valeurs utilisées (suite)

Zone dans MQMD distinct	Valeur utilisée
<i>MDBOC</i>	0
<i>MDRQ</i>	Copié à partir du descripteur de message imbriqué.
<i>MDRM</i>	Copié à partir du descripteur de message imbriqué.
<i>MDUID</i>	Copié à partir du descripteur de message imbriqué.
<i>MDACC</i>	Copié à partir du descripteur de message imbriqué.
<i>MDAID</i>	Copié à partir du descripteur de message imbriqué.
<i>MDPAT</i>	ATQM
<i>MDPAN</i>	28 premiers octets du nom du gestionnaire de files d'attente.
<i>MDPD</i>	Date à laquelle le message a été inséré dans la file d'attente de transmission.
<i>MDPT</i>	Heure à laquelle le message a été inséré dans la file d'attente de transmission.
<i>MDAOD</i>	Espaces vides
<i>MDGID</i>	GINONE
<i>MDSEQ</i>	1
<i>MDOFF</i>	0
<i>MDMFL</i>	MFNONE
<i>MDOLN</i>	ANNULER

### Zones du descripteur de message imbriqué

Les zones du descripteur de message imbriqué ont les mêmes valeurs que celles du paramètre **MSGDSC** de l'appel MQPUT ou MQPUT1, à l'exception de ce qui suit:

- La zone *MDVER* a toujours la valeur MDVER1.
- Si la zone *MDPRI* a la valeur PRQDEF, elle est remplacée par la valeur de l'attribut **DefPriority** de la file d'attente.
- Si la zone *MDPER* a la valeur PEQDEF, elle est remplacée par la valeur de l'attribut **DefPersistence** de la file d'attente.
- Si la zone *MDMID* a la valeur MINONE ou si l'option PMNMID a été spécifiée, ou si le message est un message de liste de distribution, *MDMID* est remplacé par un nouvel identificateur de message généré par le gestionnaire de files d'attente.

Lorsqu'un message de liste de distribution est divisé en messages de liste de distribution plus petits placés dans des files d'attente de transmission différentes, la zone *MDMID* de chacun des nouveaux descripteurs de message imbriqués est identique à celle du message de liste de distribution d'origine.

- Si l'option PMNCID a été spécifiée, *MDCID* est remplacé par un nouvel identificateur de corrélation généré par le gestionnaire de files d'attente.
- Les zones de contexte sont définies comme indiqué par les options PM\* spécifiées dans le paramètre **PMO**; les zones de contexte sont les suivantes:
  - *MDACC*
  - *MDAID*
  - *MDAOD*
  - *MDPAN*

- *MDPAT*
  - *MDPD*
  - *MDPT*
  - *MDUID*
- Les zones version-2 (si elles étaient présentes) sont supprimées de MQMD et déplacées dans une structure MQMDE, si une ou plusieurs des zones version-2 ont une valeur autre que la valeur par défaut.

## Insertion de messages dans des files d'attente éloignées

: Lorsqu'une application insère un message dans une file d'attente éloignée (en spécifiant directement le nom de la file d'attente éloignée ou en utilisant une définition locale de la file d'attente éloignée), le gestionnaire de files d'attente local:

- Crée une structure MQXQH contenant le descripteur de message imbriqué
- Ajoute un MQMDE si nécessaire et qu'il n'est pas déjà présent
- Ajoute les données de message d'application
- Place le message dans une file d'attente de transmission appropriée

## Insertion de messages directement dans les files d'attente de transmission

Il est également possible pour une application d'insérer un message directement dans une file d'attente de transmission. Dans ce cas, l'application doit préfixer les données de message d'application avec une structure MQXQH et initialiser les zones avec les valeurs appropriées. De plus, la zone *MDFMT* du paramètre **MSGDSC** de l'appel MQPUT ou MQPUT1 doit avoir la valeur FMXQH.

Les données de type caractère de la structure MQXQH créée par l'application doivent être dans le jeu de caractères du gestionnaire de files d'attente local (défini par l'attribut de gestionnaire de files d'attente **CodedCharSetId**) et les données de type entier doivent être dans le codage machine natif. En outre, les données de type caractère dans la structure MQXQH doivent être complétées par des blancs à la longueur définie de la zone ; les données ne doivent pas être arrêtées prématurément à l'aide d'un caractère null, car le gestionnaire de files d'attente ne convertit pas les caractères null et suivants en blancs dans la structure MQXQH.

Notez toutefois que le gestionnaire de files d'attente ne vérifie pas qu'une structure MQXQH est présente ou que des valeurs valides ont été spécifiées pour les zones.

## Obtention de messages à partir de files d'attente de transmission

Les applications qui extraient des messages d'une file d'attente de transmission doivent traiter les informations de la structure MQXQH de manière appropriée. La présence de la structure MQXQH au début des données de message d'application est indiquée par la valeur FMXQH renvoyée dans la zone *MDFMT* du paramètre **MSGDSC** de l'appel MQGET. Les valeurs renvoyées dans les zones *MDCSI* et *MDENC* du paramètre **MSGDSC** indiquent le jeu de caractères et le codage des données de type caractère et entier dans la structure MQXQH. Le jeu de caractères et le codage des données de message d'application sont définis par les zones *MDCSI* et *MDENC* dans le descripteur de message imbriqué.

## Zones

La structure MQXQH contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

### XQMD (MQMD1)

Descripteur de message d'origine.

Il s'agit du descripteur de message intégré. Il s'agit d'une copie fermée du descripteur de message MQMD qui a été spécifié en tant que paramètre **MSGDSC** dans l'appel MQPUT ou MQPUT1 lors de l'insertion initiale du message dans la file d'attente éloignée.

**Remarque :** Il s'agit d'un MQMD version-1 .

Les valeurs initiales des zones de cette structure sont identiques à celles de la structure MQMD.

### **XQRQ (chaîne de caractères de 48 octets)**

Nom de la file d'attente de destination.

Il s'agit du nom de la file d'attente de messages correspondant à la destination finale apparente du message (il peut s'avérer qu'il ne s'agit pas de la destination finale réelle si, par exemple, cette file d'attente est définie sur *XQRQM* pour être une définition locale d'une autre file d'attente éloignée).

Si le message est un message de liste de distribution (c'est-à-dire que la zone *MDFMT* du descripteur de message imbriqué est *FMDH*), *XQRQ* est vide.

La longueur de cette zone est indiquée par *LNQN*. La valeur initiale de cette zone est de 48 caractères blancs.

### **XQRQM (chaîne de caractères de 48 octets)**

Nom du gestionnaire de files d'attente de destination.

Il s'agit du nom du gestionnaire de files d'attente ou du groupe de partage de files d'attente propriétaire de la file d'attente qui est la destination finale apparente du message.

Si le message est un message de liste de distribution, *XQRQM* est vide.

La longueur de cette zone est indiquée par *LNQMN*. La valeur initiale de cette zone est de 48 caractères blancs.

### **XQSID (chaîne de caractères de 4 octets)**

Identificateur de structure.

La valeur doit être:

#### **XQSIDV**

Identificateur de la structure d'en-tête de file d'attente de transmission.

La valeur initiale de cette zone est *XQSIDV*.

### **XQVER (entier signé à 10 chiffres)**

Numéro de version de la structure.

La valeur doit être:

#### **XQVER1**

Numéro de version de la structure d'en-tête de file d'attente de transmission.

La constante suivante indique le numéro de version de la version en cours:

#### **XQVERC**

Version actuelle de la structure d'en-tête de file d'attente de transmission.

La valeur initiale de cette zone est *XQVER1*.

## **Valeurs initiales**

Nom de zone	Nom de la constante	Valeur de la constante
<i>XQSID</i>	<i>XQSIDV</i>	'XQH~'
<i>XQVER</i>	<i>XQVER1</i>	1
<i>XQRQ</i>	Aucun	Espaces vides
<i>XQRQM</i>	Aucun	Espaces vides
<i>XQMD</i>	Mêmes noms et valeurs que MQMD ; voir <a href="#">Tableau 709, à la page 1207</a>	-

Tableau 741. Zones dans MQXQH (suite)

Nom de zone	Nom de la constante	Valeur de la constante
<b>Remarques :</b>		
1. Le symbole ¬ représente un caractère blanc unique.		

## Déclaration RPG

```

D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQXQH Structure
D*
D* Structure identifier
D XQSID          1          4    INZ('XQH ')
D* Structure version number
D XQVER          5          8I 0  INZ(1)
D* Name of destination queue
D XQRQ           9          56    INZ
D* Name of destination queue manager
D XQRQM          57         104    INZ
D* Original message descriptor
D XQ1SID         105         108    INZ('MD ')
D XQ1VER         109         112I 0 INZ(1)
D XQ1REP         113         116I 0 INZ(0)
D XQ1MT          117         120I 0 INZ(8)
D XQ1EXP         121         124I 0 INZ(-1)
D XQ1FB          125         128I 0 INZ(0)
D XQ1ENC         129         132I 0 INZ(273)
D XQ1CSI         133         136I 0 INZ(0)
D XQ1FMT         137         144    INZ(' ')
D XQ1PRI         145         148I 0 INZ(-1)
D XQ1PER         149         152I 0 INZ(2)
D XQ1MID         153         176    INZ(X'00000000000000-
D                                     000000000000000000000000-
D                                     00000000000000')
D XQ1CID         177         200    INZ(X'00000000000000-
D                                     000000000000000000000000-
D                                     000000000000')
D XQ1BOC         201         204I 0 INZ(0)
D XQ1RQ          205         252    INZ
D XQ1RM          253         300    INZ
D XQ1UID         301         312    INZ
D XQ1ACC         313         344    INZ(X'00000000000000-
D                                     000000000000000000000000-
D                                     00000000000000000000-
D                                     00000000')
D XQ1AID         345         376    INZ
D XQ1PAT         377         380I 0 INZ(0)
D XQ1PAN         381         408    INZ
D XQ1PD          409         416    INZ
D XQ1PT          417         424    INZ
D XQ1AOD         425         428    INZ

```

## IBM i Appels de fonction sur IBM i

Utilisez ces informations pour en savoir plus sur les appels de fonction disponibles dans la programmation IBM i.

### Conventions utilisées dans les descriptions d'appel sous IBM i

Pour chaque appel, cette collection de rubriques fournit une description des paramètres et de l'utilisation de l'appel. Elle est suivie d'appels typiques de l'appel et de déclarations typiques de ses paramètres, dans le langage de programmation RPG.

**Important :** Lorsque vous codez des appels d'API IBM MQ, vous devez vous assurer que tous les paramètres appropriés (comme décrit dans les sections suivantes) sont fournis. Si vous ne le faites pas, vous risquez d'obtenir des résultats imprévisibles.

La description de chaque appel contient les sections suivantes:

### **Nom de l'appel**

Nom de l'appel, suivi d'une brève description de l'objet de l'appel.

### **Paramètres**

Pour chaque paramètre, le nom est suivi de son type de données entre parenthèses () et sa direction ; par exemple:

*CMPCOD* (entier décimal à 9 chiffres)-sortie

Vous trouverez plus d'informations sur les types de données de structure dans [«Types de données élémentaires»](#), à la page 1044.

La direction du paramètre peut être:

#### **Entrée**

Vous (le programmeur) devez fournir ce paramètre.

#### **Sortie**

L'appel renvoie ce paramètre.

#### **Entrée/sortie**

Vous devez fournir ce paramètre, mais il est modifié par l'appel.

Il contient également une brève description de l'objectif du paramètre, ainsi qu'une liste de toutes les valeurs que le paramètre peut prendre.

Les deux derniers paramètres de chaque appel sont un code achèvement et un code anomalie.

Le code achèvement indique si l'appel a abouti, partiellement ou pas du tout. Des informations supplémentaires sur la réussite partielle ou l'échec de l'appel sont fournies dans le code anomalie.

### **Remarques sur l'utilisation**

Informations supplémentaires sur l'appel, décrivant la façon de l'utiliser et les éventuelles restrictions liées à son utilisation.

### **Appel RPG**

Appel standard de l'appel et déclaration de ses paramètres dans RPG.

Les autres conventions de notation sont les suivantes:

### **Constantes**

Les noms des constantes sont affichés en majuscules ; par exemple, OOOOUT.

### **Tableaux**

Dans certains appels, les paramètres sont des tableaux de chaînes de caractères dont la taille n'est pas fixe. Dans les descriptions de ces paramètres, une minuscule *n* représente une constante numérique. Lorsque vous codez la déclaration pour ce paramètre, remplacez *n* par la valeur numérique requise.

## **MQBACK (Annulation des modifications) sous IBM i**

L'appel MQBACK indique au gestionnaire de files d'attente que tous les extractions et insertions de messages qui se sont produites depuis le dernier point de synchronisation doivent être annulées. Les messages insérés dans une unité d'oeuvre sont supprimés ; les messages extraits dans une unité d'oeuvre sont réintégrés dans la file d'attente.

- Cet appel est pris en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Windows

- [«Syntaxe»](#), à la page 1313



- «Remarques sur l'utilisation», à la page [1313](#)
- «Paramètres», à la page [1314](#)
- «Déclaration RPG», à la page [1315](#)

## Syntaxe

MQBACK (*Hconn, CompCode, Reason*)

## Remarques sur l'utilisation

Tenez compte de ces remarques d'utilisation lors de l'utilisation de MQBACK.

1. Cet appel peut être utilisé uniquement lorsque le gestionnaire de files d'attente lui-même coordonne l'unité d'oeuvre. Il s'agit d'une unité d'oeuvre locale, où les modifications affectent uniquement les ressources IBM MQ .
2. Dans les environnements où le gestionnaire de files d'attente ne coordonne pas l'unité d'oeuvre, l'appel d'annulation approprié doit être utilisé à la place de MQBACK. L'environnement peut également prendre en charge une interruption implicite provoquée par l'arrêt anormal de l'application.
  - Sous IBM i, cet appel peut être utilisé pour les unités d'oeuvre locales coordonnées par le gestionnaire de files d'attente. Cela signifie qu'une définition de validation ne doit pas exister au niveau du travail, c'est-à-dire que la commande STRCMTCTL avec le paramètre **CMTSCOPE (\*JOB)** ne doit pas avoir été émise pour le travail.
3. Si une application se termine avec des modifications non validées dans une unité d'oeuvre, la disposition de ces modifications varie selon que l'application se termine normalement ou anormalement. Pour plus de détails, voir les remarques sur l'utilisation dans «MQDISC (gestionnaire de files d'attente de déconnexion) sous IBM i», à la page [1353](#) .
4. Lorsqu'une application insère ou extrait des messages dans des groupes ou des segments de messages logiques, le gestionnaire de files d'attente conserve les informations relatives au groupe de messages et au message logique pour les derniers appels MQPUT et MQGET ayant abouti. Ces informations sont associées à l'identificateur de file d'attente et incluent les éléments suivants:
  - Valeurs des zones *MDGID, MDSEQ, MDOFF* et *MDMFL* dans MQMD.
  - Indique si le message fait partie d'une unité de travail.
  - Pour l'appel MQPUT: indique si le message est persistant ou non persistant.

Le gestionnaire de files d'attente conserve *trois* ensembles d'informations de groupe et de segment, un ensemble pour chacun des éléments suivants:

- Dernier appel MQPUT réussi (qui peut faire partie d'une unité de travail).
- Dernier appel MQGET réussi qui a supprimé un message de la file d'attente (il peut faire partie d'une unité d'oeuvre).
- Dernier appel MQGET ayant réussi à parcourir un message dans la file d'attente (il ne peut pas faire partie d'une unité d'oeuvre).

Si l'application insère ou obtient les messages dans le cadre d'une unité de travail, et que l'application décide alors d'en revenir à l'unité de travail, les informations de groupe et de segment sont restaurées à la valeur qu'elle avait précédemment:

- Les informations associées à l'appel MQPUT sont restaurées à la valeur qu'elles avaient avant le premier appel MQPUT réussi pour ce descripteur de file d'attente dans l'unité de travail en cours.
- Les informations associées à l'appel MQGET sont restaurées à la valeur qu'elles avaient avant le premier appel MQGET réussi pour ce descripteur de file d'attente dans l'unité d'oeuvre en cours.

Les files d'attente qui ont été mises à jour par l'application après le démarrage de l'unité de travail, mais en dehors de la portée de l'unité de travail, ne voient pas leurs informations de groupe et de segment restaurées si l'unité de travail est annulée.

La restauration des informations de groupe et de segment à sa valeur précédente lors de l'annulation d'une unité de travail permet à l'application de répartir un grand groupe de messages ou un grand message logique composé de plusieurs segments sur plusieurs unités de travail, et de redémarrer au point approprié du groupe de messages ou du message logique si l'une des unités de travail échoue. L'utilisation de plusieurs unités d'oeuvre peut s'avérer avantageuse si le gestionnaire de files d'attente local ne dispose que d'une mémoire de file d'attente limitée. Toutefois, l'application doit conserver suffisamment d'informations pour pouvoir redémarrer l'insertion ou l'obtention de messages au point approprié en cas de défaillance du système. Pour plus d'informations sur le redémarrage au point approprié après une défaillance du système, voir l'option **PMLOGO** décrite dans «**MQPMO (options d'insertion de message) sous IBM i**», à la page 1230 et l'option **GMLOGO** décrite dans «**MQGMO (options d'extraction de message) sous IBM i**», à la page 1127.

Les autres remarques d'utilisation s'appliquent uniquement lorsque le gestionnaire de files d'attente coordonne les unités de travail:

1. Une unité de travail a la même portée qu'un descripteur de connexion. Cela signifie que tous les appels IBM MQ qui affectent une unité de travail particulière doivent être exécutés à l'aide du même descripteur de connexion. Les appels émis à l'aide d'un descripteur de connexion différent (par exemple, les appels émis par une autre application) affectent une unité d'oeuvre différente. Pour plus d'informations sur la portée des descripteurs de connexion, voir le paramètre **HCONN** décrit dans «**MQCONN (Connect queue manager) sur IBM i**», à la page 1339.
2. Seuls les messages insérés ou extraits dans le cadre de l'unité d'oeuvre en cours sont affectés par cet appel.
3. Une application à exécution longue qui émet des appels MQGET, MQPUT ou MQPUT1 dans une unité d'oeuvre, mais qui n'émet jamais d'appel de validation ou d'annulation, peut provoquer le remplissage des files d'attente avec des messages qui ne sont pas disponibles pour d'autres applications. Pour éviter cette possibilité, l'administrateur doit définir l'attribut de gestionnaire de files d'attente **MaxUncommittedMsgs** sur une valeur suffisamment faible pour empêcher les applications en fuite de remplir les files d'attente, mais suffisamment élevée pour permettre aux applications de messagerie attendues de fonctionner correctement.

## Paramètres

L'appel MQBACK comporte les paramètres suivants:

### **HCONN (entier signé à 10 chiffres)-entrée**

Descripteur de connexion.

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *HCONN* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

### **CMPCOD (entier signé à 10 chiffres)-sortie**

Code achèvement.

Il peut s'agir de :

#### **CCOK**

Achèvement réussi.

#### **CCFAIL**

Echec de l'appel.

### **REASON (entier signé à 10 chiffres)-sortie**

Code anomalie qualifiant *COMCOD*.

Si *COMCOD* est CCOK:

#### **RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si *COMCOD* est CCFAIL:

**RC2219**

(2219, X'8AB') Appel MQI entré à nouveau avant la fin de l'appel précédent.

**RC2009**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**RC2018**

(2018, X'7E2') Descripteur de connexion non valide.

**RC2101**

(2101, X'835') Objet endommagé.

**RC2123**

(2123, X'84B') Le résultat de l'opération de validation ou d'exclusion est mélangé.

**RC2162**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**RC2102**

(2102, X'836') Ressources système disponibles insuffisantes.

**RC2071**

(2071, X'817') Mémoire disponible insuffisante.

**RC2195**

(2195, X'893') Une erreur inattendue s'est produite.

**Déclaration RPG**

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP          MQBACK(HCONN : COMCOD : REASON)

```

La définition de prototype pour l'appel est la suivante:

```

D*..1.....2.....3.....4.....5.....6.....7..
DMQBACK          PR          EXTPROC('MQBACK')
D* Connection handle
D HCONN          10I 0 VALUE
D* Completion code
D COMCOD          10I 0
D* Reason code qualifying COMCOD
D REASON          10I 0

```

**IBM i MQBEGIN (Début de l'unité de travail) sur IBM i**

L'appel MQBEGIN démarre une unité de travail coordonnée par le gestionnaire de files d'attente et qui peut impliquer des gestionnaires de ressources externes.

- Cet appel est pris en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Windows

- [«Syntaxe», à la page 1316](#)
- [«Remarques sur l'utilisation», à la page 1316](#)
- [«Paramètres», à la page 1317](#)
- [«Déclaration RPG», à la page 1318](#)

## Syntaxe

MQBEGIN (*HCONN, BEGOP, CMPCOD, REASON*)

## Remarques sur l'utilisation

1. L'appel MQBEGIN peut être utilisé pour démarrer une unité de travail coordonnée par le gestionnaire de files d'attente et qui peut impliquer des modifications apportées aux ressources appartenant à d'autres gestionnaires de ressources. Le gestionnaire de files d'attente prend en charge trois types d'unité de travail:

### **Gestionnaire de files d'attente-Unité d'oeuvre locale coordonnée**

Il s'agit d'une unité de travail dans laquelle le gestionnaire de files d'attente est le seul gestionnaire de ressources participant et, par conséquent, le gestionnaire de files d'attente joue le rôle de coordinateur d'unité de travail.

- Pour démarrer ce type d'unité de travail, l'option PMSYP ou GMSYP doit être spécifiée dans le premier appel MQPUT, MQPUT1 ou MQGET de l'unité de travail.

Il n'est pas nécessaire que l'application lance l'appel MQBEGIN pour démarrer l'unité de travail, mais si MQBEGIN est utilisé, l'appel se termine avec CCWARN et le code anomalie RC2121.

- Pour valider ou rétablir ce type d'unité d'oeuvre, vous devez utiliser l'appel MQCMIT ou MQBACK.

### **Unité d'oeuvre globale coordonnée par le gestionnaire de files d'attente**

Il s'agit d'une unité de travail dans laquelle le gestionnaire de files d'attente agit en tant que coordinateur d'unité de travail, à la fois pour les IBM MQ ressources *et* pour les ressources appartenant à d'autres gestionnaires de ressources. Ces gestionnaires de ressources coopèrent avec le gestionnaire de files d'attente pour s'assurer que toutes les modifications apportées aux ressources de l'unité d'oeuvre sont validées ou annulées ensemble.

- Pour démarrer ce type d'unité de travail, l'appel MQBEGIN doit être utilisé.
- Pour valider ou dévalider ce type d'unité d'oeuvre, les appels MQCMIT et MQBACK doivent être utilisés.

### **Unité de travail globale coordonnée à l'extérieur**

Il s'agit d'une unité de travail dans laquelle le gestionnaire de files d'attente est un participant, mais le gestionnaire de files d'attente n'agit pas en tant que coordinateur d'unité de travail. Au lieu de cela, il existe un coordinateur d'unité de travail externe avec lequel le gestionnaire de files d'attente coopère.

- Pour démarrer ce type d'unité de travail, l'appel correspondant fourni par le coordinateur externe de l'unité de travail doit être utilisé.

Si l'appel MQBEGIN est utilisé pour tenter de démarrer l'unité de travail, l'appel échoue avec le code anomalie RC2012.

- Pour valider ou mettre hors service ce type d'unité de travail, les appels de validation et de mise hors service fournis par le coordinateur d'unité de travail externe doivent être utilisés.

Si l'appel MQCMIT ou MQBACK est utilisé pour tenter de valider ou d'éliminer l'unité de travail, l'appel échoue avec le code anomalie RC2012.

2. Si l'application se termine avec des modifications non validées dans une unité de travail, la disposition de ces modifications varie selon que l'application se termine normalement ou anormalement. Pour plus de détails, voir les remarques sur l'utilisation dans [«MQDISC \(gestionnaire de files d'attente de déconnexion\) sous IBM i»](#), à la page 1353 .
3. Une application ne peut participer qu'à une seule unité de travail à la fois. L'appel MQBEGIN échoue avec le code anomalie RC2128 s'il existe déjà une unité de travail pour l'application, quel que soit le type d'unité de travail.
4. L'appel MQBEGIN n'est pas valide dans un environnement client IBM MQ . Une tentative d'utilisation de l'appel échoue avec le code anomalie RC2012.

5. Lorsque le gestionnaire de files d'attente joue le rôle de coordinateur d'unité de travail pour les unités de travail globales, les gestionnaires de ressources qui peuvent participer à l'unité de travail sont définis dans le fichier de configuration du gestionnaire de files d'attente.
6. Sous IBM i, les trois types d'unité de travail sont pris en charge comme suit:
  - Les **unités d'oeuvre locales coordonnées par le gestionnaire de files d'attente** ne peuvent être utilisées que lorsqu'une définition de validation n'existe pas au niveau du travail, c'est-à-dire que la commande STRCMTCTL avec le paramètre **CMTSCOPE (\*JOB)** ne doit pas avoir été émise pour le travail.
  - Les **unités d'oeuvre globales coordonnées par le gestionnaire de files d'attente** ne sont pas prises en charge.
  - Les **unités d'oeuvre globales coordonnées en externe** ne peuvent être utilisées que lorsqu'une définition de validation existe au niveau du travail, c'est-à-dire que la commande STRCMTCTL avec le paramètre **CMTSCOPE (\*JOB)** doit avoir été émise pour le travail. Si cela a été fait, les opérations IBM i COMMIT et ROLLBACK s'appliquent aux ressources IBM MQ ainsi qu'aux ressources appartenant à d'autres gestionnaires de ressources participants.

## Paramètres

L'appel MQBEGIN comporte les paramètres suivants:

### **HCONN (entier signé à 10 chiffres)-entrée**

Descripteur de connexion.

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *HCONN* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

### **BEGOP (MQBO)-entrée/sortie**

Options qui contrôlent l'action de MQBEGIN.

Pour plus d'informations, voir «MQBO (options de début) sous IBM i», à la page 1067.

Si aucune option n'est requise, les programmes écrits en assembleur C ou S/390 peuvent spécifier une adresse de paramètre null au lieu de spécifier l'adresse d'une structure MQBO.

### **CMPCOD (entier signé à 10 chiffres)-sortie**

Code achèvement.

Il peut s'agir de :

#### **CCOK**

Achèvement réussi.

#### **CCWARN**

Avertissement (achèvement partiel).

#### **CCFAIL**

Echec de l'appel.

### **REASON (entier signé à 10 chiffres)-sortie**

Code anomalie qualifiant *CMPCOD*.

Si *CMPCOD* est CCOK:

#### **RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CMPCOD* est CCWARN:

#### **RC2121**

(2121, X'849') Aucun gestionnaire de ressources participant enregistré.

#### **RC2122**

(2122, X'84A') Gestionnaire de ressources participant non disponible.

Si *CMPCOD* est CCFAIL:

**RC2134**

(2134, X'856') Structure des options de début incorrecte.

**RC2219**

(2219, X'8AB') Appel MQI entré à nouveau avant la fin de l'appel précédent.

**RC2009**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**RC2012**

(2012, X'7DC') Appel non valide dans l'environnement.

**RC2018**

(2018, X'7E2') Descripteur de connexion non valide.

**RC2046**

(2046, X'7FE') Options incorrectes ou incohérentes.

**RC2162**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**RC2102**

(2102, X'836') Ressources système disponibles insuffisantes.

**RC2071**

(2071, X'817') Mémoire disponible insuffisante.

**RC2195**

(2195, X'893') Une erreur inattendue s'est produite.

**RC2128**

(2128, X'850') Unité de travail déjà démarrée.

## Déclaration RPG

```
C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQBEGIN(HCONN : BEGOP : CMPCOD :
C                      REASON)
```

La définition de prototype pour l'appel est la suivante:

```
D*..1.....2.....3.....4.....5.....6.....7..
DMQBEGIN      PR          EXTPROC('MQBEGIN')
D* Connection handle
D HCONN          10I 0 VALUE
D* Options that control the action of MQBEGIN
D BEGOP          12A
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CMPCOD
D REASON          10I 0
```

## MQBUFMH (Convertir la mémoire tampon en descripteur de message) sous IBM i

L'appel de fonction MQBUFMH convertit une mémoire tampon en descripteur de message et est l'inverse de l'appel MQMHBUF.

Cet appel prend un descripteur de message et des propriétés MQRFH2 dans la mémoire tampon et les rend disponibles via un descripteur de message. Les propriétés MQRFH2 des données de message sont éventuellement supprimées. Les zones *Encoding*, *CodedCharSetIdet Format* du descripteur de message sont mises à jour, si nécessaire, pour décrire correctement le contenu de la mémoire tampon après la suppression des propriétés.

- [«Syntaxe», à la page 1319](#)

- «Remarques sur l'utilisation», à la page [1319](#)
- «Paramètres», à la page [1319](#)
- «Déclaration RPG», à la page [1321](#)

## Syntaxe

MQBUFMH (*Hconn*, *Hmsg*, *BufMsgHOpts*, *MsgDesc*, *Buffer*, *BufferLength*, *DataLength*, *CompCode*, *Reason*)

## Remarques sur l'utilisation

Les appels MQBUFMH ne peuvent pas être interceptés par des exits API-une mémoire tampon est convertie en descripteur de message dans l'espace d'application ; l'appel n'atteint pas le gestionnaire de files d'attente.

## Paramètres

L'appel MQBUFMH comporte les paramètres suivants:

### HCONN (entier signé à 10 chiffres)-entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *HCONN* doit correspondre au descripteur de connexion qui a été utilisé pour créer le descripteur de message spécifié dans le paramètre **Hmsg**.

Si le descripteur de message a été créé à l'aide de HCUNAS, une connexion valide doit être établie sur l'unité d'exécution qui convertit une mémoire tampon en descripteur de message. Si une connexion valide n'est pas établie, l'appel échoue avec RC2009.

### HMSG (entier signé à 20 chiffres)-entrée

Ce descripteur est le descripteur de message pour lequel une mémoire tampon est requise. La valeur a été renvoyée par un appel MQCRTMH précédent.

### BMHOPT (MQBMHO)-entrée

La structure MQBMHO permet aux applications de spécifier des options qui contrôlent la façon dont les descripteurs de message sont produits à partir des mémoires tampon.

Pour plus d'informations, voir «[MQBMHO \(Buffer to message handle options\) sur IBM i](#)», à la page [1065](#).

### MSGDSC (MQMD)-entrée / sortie

La structure *MSGDSC* contient les propriétés de descripteur de message et décrit le contenu de la zone tampon.

En sortie de l'appel, les propriétés sont éventuellement supprimées de la zone tampon et, dans ce cas, le descripteur de message est mis à jour pour décrire correctement la zone tampon.

Les données de cette structure doivent être dans le jeu de caractères et le codage de l'application.

### BUFLEN (entier signé à 10 chiffres)-entrée

*BUFLEN* est la longueur de la zone tampon, en octets.

Une valeur *BUFLEN* de zéro octet est valide et indique que la zone tampon ne contient aucune donnée.

### BUFFER (chaîne de 1 octet x BUFLEN)-entrée / sortie

*BUFFER* définit la zone contenant la mémoire tampon de messages. Pour la plupart des données, vous devez aligner la mémoire tampon sur une limite de 4 octets.

Si *BUFFER* contient des données de type caractère ou numérique, définissez les zones *CodedCharSetId* et *Encoding* du paramètre **MSGDSC** sur les valeurs appropriées aux données ; cela permet la conversion des données, si nécessaire.

Si des propriétés sont trouvées dans la mémoire tampon de messages, elles sont éventuellement supprimées ; elles sont ensuite disponibles à partir du descripteur de message lors du retour de l'appel.

Dans le langage de programmation C, le paramètre est déclaré comme un pointeur vers vide, ce qui signifie que l'adresse de tout type de données peut être spécifiée comme paramètre.

Si le paramètre **BUFLEN** a pour valeur zéro, il n'est pas fait référence à *BUFFER* . Dans ce cas, l'adresse de paramètre transmise par les programmes écrits en assembleur C ou System/390 peut être null.

#### **DATLEN (entier signé à 10 chiffres)-sortie**

*DATLEN* est la longueur, en octets, de la mémoire tampon sur laquelle les propriétés peuvent être supprimées.

#### **CMPCOD (entier signé à 10 chiffres)-sortie**

##### **CCOK**

Achèvement réussi.

##### **CCFAIL**

Echec de l'appel.

#### **REASON (entier signé à 10 chiffres)-sortie**

Code anomalie qualifiant *CMPCOD*.

Si *CMPCOD* est CCOK:

##### **RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CMPCOD* est CCFAIL:

##### **RC2204**

(2204, X'089C') Carte non disponible.

##### **RC2130**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

##### **RC2157**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

##### **RC2489**

(2489, X'09B9') La structure des options de la mémoire tampon vers le descripteur de message n'est pas valide.

##### **RC2004**

(2004, X'07D4') Paramètre de mémoire tampon non valide.

##### **RC2005**

(2005, X'07D5') Paramètre de longueur de mémoire tampon non valide.

##### **RC2219**

(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

##### **RC2009**

(2009, X'07D9') La connexion au gestionnaire de files d'attente a été perdue.

##### **RC2460**

(2460, X'099C') Descripteur de message non valide.

##### **RC2026**

(2026, X'07EA') Descripteur de message non valide.

##### **RC2499**

(2499, X'09C3') Descripteur de message déjà utilisé.

##### **RC2046**

(2046, X'07FE') Options non valides ou non cohérentes.

##### **RC2334**

(2334, X'091E') Structure MQRFH2 non valide.



## RC2421

(2421, X'0975') Un dossier MQRFH2 contenant des propriétés n'a pas pu être analysé.

## RC2195

(2195, X'893') Une erreur inattendue s'est produite.

## Déclaration RPG

```
C*.1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQBUFMH(HCONN : HMSG : BMHOPT :
                          MSGDSC : BUFLen : BUFFER :
                          DATLEN : CMPCOD : REASON)
```

La définition de prototype pour l'appel est la suivante:

```
DMQBUFMH      PR          EXTPROC('MQBUFMH')
D* Connection handle
D HCONN              10I 0
D* Message handle
D HMSG              10I 0
D* Options that control the action of MQBUFMH
D BMHOPT            12A VALUE
D* Message descriptor
D MSGDSC              364A
D* Length in bytes of the Buffer area
D BUFLen            10I 0
D* Area to contain the message buffer
D BUFFER            * VALUE
D* Length of the output buffer
D DATLEN            10I 0
D* Completion code
D CMPCOD            10I 0
D* Reason code qualifying CompCode
D REASON            10I 0
```

## IBM i MQCB (Manage callback) sur IBM i

L'appel MQCB réenregistre un rappel pour le descripteur d'objet spécifié et contrôle l'activation et les modifications apportées au rappel.

Un rappel est un élément de code (spécifié comme le nom d'une fonction qui peut être liée dynamiquement ou comme un pointeur de fonction) qui est appelé par IBM MQ lorsque certains événements se produisent.

Pour utiliser MQCB et MQCTL sur un client V7, vous devez être connecté à un serveur V7 et le paramètre **SHARECNV** du canal doit avoir une valeur différente de zéro.

Pour plus d'informations sur les unités de travail globales, voir: [Unités de travail globales](#).

Les types de rappel pouvant être définis sont les suivants:

### Consommateur de message

Une fonction de rappel de consommateur de message est appelée lorsqu'un message répondant aux critères de sélection spécifiés est disponible sur un descripteur d'objet.

Une seule fonction de rappel peut être enregistrée pour chaque descripteur d'objet. Si une seule file d'attente doit être lue avec plusieurs critères de sélection, la file d'attente doit être ouverte plusieurs fois et une fonction de consommateur doit être enregistrée sur chaque descripteur.

### Gestionnaire d'événements

Le gestionnaire d'événements est appelé pour les conditions qui affectent l'ensemble de l'environnement de rappel.

La fonction est appelée lorsqu'une condition d'événement se produit, par exemple, l'arrêt ou la mise au repos d'un gestionnaire de files d'attente ou d'une connexion.

La fonction n'est pas appelée pour les conditions qui sont spécifiques à un consommateur de message unique, par exemple RC2016; elle est toutefois appelée si une fonction de rappel ne se termine pas normalement.

- «Syntaxe», à la page 1322
- «Remarques sur l'utilisation de MQCB», à la page 1322
- «Paramètres de MQCB», à la page 1323
- «Déclaration RPG», à la page 1329

## Syntaxe

MQCB (HCONN, OPERATN, HOBJ, CBDSC, MSGDSC, GMO, CMPCOD, REASON)

## Remarques sur l'utilisation de MQCB

1. MQCB permet de définir l'action à appeler pour chaque message, correspondant aux critères spécifiés, disponible dans la file d'attente. Lorsque l'action est traitée, le message est supprimé de la file d'attente et transmis au consommateur de message défini, ou un jeton de message est fourni, qui est utilisé pour extraire le message.
2. MQCB peut être utilisé pour définir des routines de rappel avant de démarrer la consommation avec MQCTL ou à partir d'une routine de rappel.
3. Pour utiliser MQCB en dehors d'une routine de rappel, vous devez d'abord interrompre la consommation des messages à l'aide de MQCTL et reprendre la consommation par la suite.

## Séquence de rappel de consommateur de message

Vous pouvez configurer un destinataire pour qu'il appelle le rappel à des points clés au cours du cycle de vie du destinataire. Exemple :

- lorsque le consommateur est enregistré pour la première fois,
- lorsque la connexion est démarrée,
- lorsque la connexion est arrêtée et
- lorsque le consommateur est désenregistré, soit explicitement, soit implicitement par un MQCLOSE.

Verbe	Explication
MQCTL (DEMARRAGE)	Appel MQCTL à l'aide de l'opération CTLSR
MQCTL (STOP)	Appel MQCTL à l'aide de l'opération CTLSP
MQCTL (en attente)	Appel MQCTL à l'aide de l'opération CTLSW

Permet au consommateur de conserver l'état associé au consommateur. Lorsqu'un rappel est demandé par une application, les règles d'appel du consommateur sont les suivantes:

### S'INSCRIRE

Est toujours le premier type d'appel du rappel.

Est toujours appelé sur la même unité d'exécution que l'appel MQCB (CBREG).

### DEBUT

Est toujours appelé de manière synchrone avec l'instruction MQCTL (START).

- Tous les rappels START sont terminés avant que l'instruction MQCTL (START) ne soit renvoyée.

Se trouve sur la même unité d'exécution que la distribution des messages si CTLTHR est demandé.

L'appel avec démarrage n'est pas garanti si, par exemple, un rappel précédent émet MQCTL (STOP) pendant MQCTL (START).

## **ARRETER**

Aucun autre message ou événement n'est distribué après cet appel tant que la connexion n'est pas redémarrée.

Une commande STOP est garantie si l'application a été précédemment appelée pour la commande START, un message ou un événement.

## **ANNULER l'enregistrement**

Est toujours le dernier type d'appel du rappel.

Assurez-vous que votre application effectue une initialisation et un nettoyage basés sur les unités d'exécution dans les rappels START et STOP. Vous pouvez effectuer une initialisation et un nettoyage non basés sur des unités d'exécution avec les rappels REGISTER et REGISISTER.

Ne faites aucune hypothèse sur la durée de vie et la disponibilité de l'unité d'exécution autre que celle qui est indiquée. Par exemple, ne vous fiez pas à une unité d'exécution qui reste active après le dernier appel à REGISTERISTER. De même, lorsque vous avez choisi de ne pas utiliser CTLTHR, ne supposez pas que l'unité d'exécution existe à chaque fois que la connexion est démarrée.

Si votre application a des exigences particulières en matière de caractéristiques d'unité d'exécution, elle peut toujours créer une unité d'exécution en conséquence, puis utiliser MQCTL (WAIT). Cette étape *donne* l'unité d'exécution à IBM MQ pour la distribution asynchrone des messages.

## **Utilisation de la connexion de consommateur de message**

Normalement, lorsqu'une application émet un autre appel MQI alors qu'un appel est en attente, l'appel échoue avec le code anomalie RC2219.

Il existe cependant des cas particuliers où l'application doit émettre un autre appel MQI avant que l'appel précédent ne soit terminé. Par exemple, le consommateur peut être appelé lors d'un appel MQCB avec CBRE.

Dans une telle instance, lorsque, suite à l'émission par l'application d'une instruction MQCB ou MQCTL, l'application est rappelée, l'application est autorisée à émettre un autre appel MQI. Cette instance signifie que vous pouvez émettre, par exemple, un appel MQOPEN dans la fonction de consommateur lorsqu'elle est appelée avec un type CBCCALLT CBCTRC. Tout appel MQI, à l'exception de MQDISC, est autorisé.

## **Paramètres de MQCB**

L'appel MQCB comporte les paramètres suivants:

### **HCONN (entier signé à 10 chiffres)-entrée**

Fonction de rappel de gestion-Paramètre HCONN.

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *HCONN* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

### **OPERATN (entier signé à 10 chiffres)-entrée**

Fonction de gestion de rappel-Paramètre OPERATN.

Opération en cours de traitement sur le rappel défini pour le descripteur d'objet spécifié. Vous devez spécifier l'une des options suivantes ; si plusieurs options sont requises, les valeurs peuvent être ajoutées (n'ajoutez pas la même constante plusieurs fois) ou combinées à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations de bit).

Les combinaisons qui ne sont pas valides sont notées ; toutes les autres combinaisons sont valides.

### **CREG**

Définissez la fonction de rappel pour le descripteur d'objet spécifié. Cette opération définit la fonction à appeler et les critères de sélection à utiliser.

Si une fonction de rappel est déjà définie pour le descripteur d'objet, la définition est remplacée. Si une erreur est détectée lors du remplacement du rappel, la fonction est désenregistrée.

Si un rappel est enregistré dans la même fonction de rappel dans laquelle il a été précédemment désenregistré, il est traité comme une opération de remplacement ; les appels initiaux ou finaux ne sont pas appelés.

Vous pouvez utiliser CBREG avec CTLSU ou CTLRE.

### **CUNR**

Arrête la consommation de messages pour le descripteur d'objet et supprime le descripteur de ceux qui sont éligibles pour un rappel.

Un rappel est automatiquement désenregistré si le descripteur associé est fermé.

Si CBUNR est appelé à partir d'un consommateur et que le rappel a un appel d'arrêt défini, il est appelé lors du retour du consommateur.

Si cette opération est émise sur un *Hobj* sans destinataire enregistré, l'appel est renvoyé avec RC2448.

### **CTLSU**

Interrompt la consommation de messages pour le descripteur d'objet.

Si cette opération est appliquée à un gestionnaire d'événements, le gestionnaire d'événements n'obtient pas d'événements lorsqu'il est suspendu et les événements manqués lorsqu'il est à l'état suspendu ne sont pas fournis à l'opération lors de sa reprise.

En cas d'interruption, la fonction de consommateur continue d'obtenir les rappels de type de contrôle.

### **CTLRE**

Reprenez la consommation de messages pour le descripteur d'objet.

Si cette opération est appliquée à un gestionnaire d'événements, le gestionnaire d'événements n'obtient pas d'événements lorsqu'il est suspendu et les événements manqués lorsqu'il est à l'état suspendu ne sont pas fournis à l'opération lors de sa reprise.

### **CBDSC (MQCBD)-entrée**

Fonction de gestion de rappel-Paramètre CBDSC.

Il s'agit d'une structure qui identifie la fonction de rappel qui est enregistrée par l'application et les options utilisées lors de son enregistrement.

Pour plus de détails sur la structure, voir «MQCBD-Descripteur de rappel», à la page 297 .

Le descripteur de rappel est requis uniquement pour l'option CBREG ; si le descripteur n'est pas requis, l'adresse de paramètre transmise peut être null.

### **HOBJ (entier signé à 10 chiffres)-entrée**

Fonction de gestion de rappel-Paramètre HOBJ.

Ce descripteur représente l'accès qui a été établi à l'objet à partir duquel un message doit être consommé. Il s'agit d'un descripteur qui a été renvoyé à partir d'un précédent appel MQOPEN ou MQSUB (dans le paramètre **HOBJ** ).

*HOBJ* n'est pas requis lors de la définition d'une routine de gestionnaire d'événements (CBTEH) et doit être spécifié en tant que HONONE.

Si ce *Hobj* a été renvoyé à partir d'un appel MQOPEN, la file d'attente doit avoir été ouverte avec une ou plusieurs des options suivantes:

- OOINPS
- OOINPX
- OOINPQ
- OOBROW

## MSGDSC (MQMD)-entrée

Paramètre -MSGDSC de la fonction de rappel de gestion.

Cette structure décrit les attributs du message requis ainsi que ceux du message récupéré.

Le paramètre **MsgDesc** définit les attributs des messages requis par le consommateur et la version du MQMD à transmettre au consommateur de message.

*MsgId*, *CorrelId*, *GroupId*, *MsgSeqNumber* et *Offset* dans le MQMD sont utilisés pour la sélection des messages, en fonction des options spécifiées dans le paramètre **GetMsgOpts**.

*Encoding* et *CodedCharSetId* sont utilisés pour la conversion de message si vous spécifiez l'option GMCONV.

Pour plus de détails, voir [MQMD](#).

*MsgDesc* est utilisé uniquement pour CBREG et, si vous avez besoin de valeurs autres que la valeur par défaut pour les zones, *MsgDesc* n'est pas utilisé pour un gestionnaire d'événements.

Si le descripteur n'est pas requis, l'adresse de paramètre transmise peut être null.

Notez que si plusieurs consommateurs sont enregistrés dans la même file d'attente avec des sélecteurs qui se chevauchent, le consommateur choisi pour chaque message n'est pas défini.

## OGM (MQGMO)-entrée

Gestion de la fonction de rappel-Paramètre GMO.

Options qui contrôlent la façon dont le consommateur de message obtient les messages.

Toutes les options ont la signification décrite dans «[MQGMO \(options d'extraction de message\) sous IBM](#)», à la page 1127, lorsqu'elles sont utilisées sur un appel MQGET, sauf:

### GMSSIG

Cette option n'est pas autorisée.

### GMBRWF, GMBRWN, GMMBH, GMMBC

L'ordre des messages distribués à un consommateur de navigation est dicté par les combinaisons de ces options. Les combinaisons significatives sont les suivantes:

#### GMBRWF

Le premier message de la file d'attente est distribué de manière répétée au consommateur. Cela est utile lorsque le consommateur consomme de façon destructive le message dans le rappel. Utilisez cette option avec précaution.

#### GMBRWN

Le destinataire reçoit chaque message de la file d'attente, de la position actuelle du curseur jusqu'à la fin de la file d'attente.

#### GMBRWF + GMBRWN

Le curseur est réinitialisé au début de la file d'attente. Le destinataire reçoit ensuite chaque message jusqu'à ce que le curseur atteigne la fin de la file d'attente.

#### GMBRWF + GMMBH ou GMMBC

A partir du début de la file d'attente, le consommateur reçoit le premier message non marqué de la file d'attente, qui est ensuite marqué pour ce consommateur. Cette combinaison permet de s'assurer que le consommateur peut recevoir de nouveaux messages ajoutés derrière le point de curseur en cours.

#### GMBRWN + GMMBH ou GMMBC

A partir de la position du curseur, le consommateur reçoit le message non marqué suivant dans la file d'attente, qui est ensuite marqué pour ce consommateur. Utilisez cette combinaison avec précaution car des messages peuvent être ajoutés à la file d'attente derrière la position actuelle du curseur.

#### GMBRWF + GMBRWN + GMMBH ou GMMBC

Cette combinaison n'est pas autorisée, si elle est utilisée, l'appel renvoie RC2046.

### **GMNWT, GMWT et GMWI**

Ces options contrôlent la façon dont le consommateur est appelé.

#### **GMNWT**

Le consommateur n'est jamais appelé avec RC2033. Le consommateur est appelé uniquement pour les messages et les événements

#### **GMWT avec un GMWI nul**

Le code RC2033 est transmis au consommateur uniquement lorsqu'il n'y a pas de message et

- le consommateur a été démarré
- le consommateur a reçu au moins un message depuis le dernier code raison d'absence de message.

Cela empêche le consommateur de s'interroger dans une boucle occupée lorsqu'un intervalle d'attente de zéro est spécifié.

#### **GMWT et un GMWI positif**

L'utilisateur est appelé après l'intervalle d'attente spécifié avec le code anomalie RC2033. Cet appel est effectué indépendamment du fait que des messages aient été distribués ou non au consommateur. Cela permet à l'utilisateur d'effectuer un traitement de signal de présence ou de type de lot.

#### **GMWT et GMWI de WIULIM**

Indique une attente infinie avant de renvoyer RC2033. Le consommateur n'est jamais appelé avec RC2033.

*GMO* est utilisé uniquement pour CBREG et, si vous avez besoin de valeurs autres que la valeur par défaut pour les zones, *GMO* n'est pas utilisé pour un gestionnaire d'événements.

Si les options ne sont pas requises, l'adresse de paramètre transmise peut être null.

Si un descripteur de propriétés de message est fourni dans la structure MQGMO, une copie est fournie dans la structure MQGMO qui est transmise au rappel de consommateur. En cas de retour de l'appel MQCB, l'application peut supprimer le descripteur de propriétés de message.

### **CMPCOD (entier signé à 10 chiffres)-sortie**

Fonction de gestion des rappels-Paramètre CMPCOD.

Code achèvement ; il s'agit d'un des codes suivants :

#### **CCOK**

Achèvement réussi.

#### **CCWARN**

Avertissement (achèvement partiel).

#### **CCFAIL**

Echec de l'appel.

### **REASON (entier signé à 10 chiffres)-sortie**

Fonction de gestion des rappels-Paramètre REASON.

Les codes anomalie suivants sont ceux que le gestionnaire de files d'attente peut renvoyer pour le paramètre **REASON** .

Si *CMPCOD* est CCOK:

#### **RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est CCFAIL:

#### **RC2204**

(2204, X'89C') Adaptateur non disponible.

#### **RC2133**

(2133, X'855') Impossible de charger les modules de service de conversion de données.

- RC2130**  
(2130, X'852') Impossible de charger le module de service d'adaptateur.
- RC2374**  
(2374, X'946') Echec de l'exit API.
- RC2183**  
(2183, X'887') Impossible de charger l'exit API.
- RC2157**  
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.
- RC2005**  
(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.
- RC2219**  
(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.
- RC2487**  
(2487, X'9B7') Zone de type de rappel incorrecte.
- RC2448**  
(2448, X' 990') Impossible de désenregistrer, de suspendre ou de reprendre car il n'y a pas de rappel enregistré.
- RC2486**  
(2486, X'9B6') *CallbackFunction* ou *CallbackName* doit être spécifié, mais pas les deux.
- RC2483**  
(2483, X'9B3') Zone de type de rappel incorrecte.
- RC2484**  
(2484, X'9B4') Zone d'options MQCBD incorrecte.
- RC2140**  
(2140, X'85C') Demande d'attente rejetée par CICS.
- RC2009**  
(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.
- RC2217**  
(2217, X'8A9') Non autorisé pour la connexion.
- RC2202**  
(2202, X'89A') Connexion en cours de mise au repos.
- RC2203**  
(2203, X'89B') Connexion en cours d'arrêt.
- RC2207**  
(2207, X'89F') Erreur d'identificateur de corrélation.
- RC2010**  
(2010, X'7DA') Paramètre de longueur des données non valide.
- RC2016**  
(2016, X'7E0') Se bloque pour la file d'attente.
- RC2351**  
(2351, X'92F') Conflit entre les unités d'oeuvre globales.
- RC2186**  
(2186, X'88A') Structure des options de message Get non valide.
- RC2353**  
(2353, X'931') Descripteur en cours d'utilisation pour l'unité d'oeuvre globale.
- RC2018**  
(2018, X'7E2') Descripteur de connexion non valide.
- RC2019**  
(2019, X'7E3') Descripteur d'objet non valide.

- RC2259**  
(2259, X'8D3') Spécification de navigation incohérente.
- RC2245**  
(2245, X'8C5') Spécification d'unité d'oeuvre incohérente.
- RC2246**  
(2246, X'8C6') Message sous le curseur non valide pour la récupération.
- RC2352**  
(2352, X'930') Conflit entre l'unité d'oeuvre globale et l'unité d'oeuvre locale.
- RC2247**  
(2247, X'8C7') Options de correspondance non valides.
- RC2485**  
(2485, X'9B4') Zone *MaxMsgLength* incorrecte.
- RC2026**  
(2026, X'7EA') Descripteur de message non valide.
- RC2497**  
(2497, X'9C1') Le point d'entrée de fonction spécifié est introuvable dans le module.
- RC2496**  
(2496, X'9C0') Le module a été trouvé, mais son type est incorrect ; il ne s'agit pas d'un module 32 bits, 64 bits ou d'une bibliothèque de liens dynamiques valide.
- RC2495**  
(2495, X'9BF') Module introuvable dans le chemin de recherche ou non autorisé à être chargé.
- RC2250**  
(2250, X'8CA') Numéro de séquence de message non valide.
- RC2331**  
(2331, X'91B') Utilisation du jeton de message non valide.
- RC2033**  
(2033, X'7F1') Aucun message disponible.
- RC2034**  
(2034, X'7F2') Curseur d'exploration non positionné sur le message.
- RC2036**  
(2036, X'7F4') File d'attente non ouverte pour exploration.
- RC2037**  
(2037, X'7F5') File d'attente non ouverte pour saisie.
- RC2041**  
(2041, X'7F9') Définition d'objet modifiée depuis son ouverture.
- RC2101**  
(2101, X'835') Objet endommagé.
- RC2206**  
(2206, X'89E') Code d'opération incorrect sur l'appel API.
- RC2046**  
(2046, X'7FE') Options incorrectes ou incohérentes.
- RC2193**  
(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.
- RC2052**  
(2052, X'804') File d'attente supprimée.
- RC2394**  
(2394, X'95A') Type d'index incorrect pour la file d'attente.
- RC2058**  
(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.



**RC2059**

(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

**RC2161**

(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.

**RC2162**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**RC2102**

(2102, X'836') Ressources système disponibles insuffisantes.

**RC2069**

(2069, X'815') Signal en attente pour ce descripteur.

**RC2071**

(2071, X'817') Mémoire disponible insuffisante.

**RC2109**

(2109, X'83D') Appel supprimé par le programme d'exit.

**RC2024**

(2024, X'7E8') Aucun autre message pouvant être traité au sein de l'unité d'oeuvre en cours.

**RC2072**

(2072, X'818') Prise en charge du point de synchronisation non disponible.

**RC2195**

(2195, X'893') Une erreur inattendue s'est produite.

**RC2354**

(2354, X'932') Echec de l'inscription dans l'unité d'oeuvre globale.

**RC2355**

(2355, X'933') Mélange d'appels d'unité d'oeuvre non pris en charge.

**RC2255**

(2255, X'8CF') Unité d'oeuvre non disponible pour le gestionnaire de files d'attente à utiliser.

**RC2090**

(2090, X'82A') Intervalle d'attente non valide dans MQGMO.

**RC2256**

(2256, X'8D0') Version MQGMO fournie non valide.

**RC2257**

(2257, X'8D1') Version MQMD fournie non valide.

**RC2298**

(2298, X'8FA') La fonction demandée n'est pas disponible dans l'environnement en cours.

**Déclaration RPG**

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQCB(HCONN : OPERATN : CBDSC :
                      HOBJ : MSGDSC : GMO :
                      DATLEN : CMPCOD : REASON)

```

La définition de prototype pour l'appel est la suivante:

```

DMQCB          PR          EXTPROC('MQCB')
D* Connection handle
D HCONN          10I 0 VALUE
D* Operation
D OPERATN        10I 0 VALUE
D* Callback descriptor
D CBDSC          180A
D* Object handle
D HOBJ           10I 0 VALUE
D* Message Descriptor
D MSGDSC        364A

```

```

D* Get options
D GMO 112A
D* Completion code
D CMPCOD 10I 0
* Reason code qualifying CompCode
D REASON 10I 0

```

## MQCLOSE (Fermer l'objet) sur IBM i

L'appel MQCLOSE abandonne l'accès à un objet et est l'inverse de l'appel MQOPEN.

- «Syntaxe», à la page [1330](#)
- «Remarques sur l'utilisation», à la page [1330](#)
- «Paramètres», à la page [1331](#)
- «Déclaration RPG», à la page [1336](#)

### Syntaxe

MQCLOSE (*HCONN, HOBJ, OPTS, CMPCOD, REASON*)

### Remarques sur l'utilisation

1. Lorsqu'une application émet l'appel MQDISC ou se termine normalement ou anormalement, tous les objets qui ont été ouverts par l'application et qui sont toujours ouverts sont fermés automatiquement avec l'option CONONE.
2. Les points suivants s'appliquent si l'objet en cours de fermeture est une *file d'attente*:
  - Si des opérations sur la file d'attente sont effectuées dans le cadre d'une unité de travail, la file d'attente peut être fermée avant ou après l'exécution du point de synchronisation sans affecter le résultat du point de synchronisation.
  - Si la file d'attente a été ouverte avec l'option OOBROW, le curseur de navigation est détruit. Si la file d'attente est rouverte ultérieurement avec l'option OOBROW, un nouveau curseur de navigation est créé (voir l'option OOBROW décrite dans MQOPEN).
  - Si un message est actuellement verrouillé pour cet identificateur au moment de l'appel MQCLOSE, le verrou est libéré (voir l'option GMLK décrite dans [«MQGMO \(options d'extraction de message\) sous IBM i»](#), à la page [1127](#)).
3. Les points suivants s'appliquent si l'objet en cours de fermeture est une *file d'attente dynamique* (permanente ou temporaire):
  - Pour une file d'attente dynamique, les options CODEL ou COPURG peuvent être spécifiées quelles que soient les options indiquées dans l'appel MQOPEN correspondant.
  - Lorsqu'une file d'attente dynamique est supprimée, tous les appels MQGET avec l'option GMWT qui sont en attente dans la file d'attente sont annulés et le code anomalie RC2052 est renvoyé. Voir l'option GMWT décrite dans [«MQGMO \(options d'extraction de message\) sous IBM i»](#), à la page [1127](#).

Une fois qu'une file d'attente dynamique a été supprimée, tout appel (autre que MQCLOSE) qui tente de référencer la file d'attente à l'aide d'un descripteur *HOBJ* précédemment acquis échoue avec le code anomalie RC2052.

Sachez que même si une file d'attente supprimée n'est pas accessible par les applications, elle n'est pas supprimée du système et les ressources associées ne sont pas libérées tant que tous les descripteurs qui font référence à la file d'attente n'ont pas été fermés et que toutes les unités de travail qui affectent la file d'attente n'ont pas été validées ou annulées.

- Lorsqu'une file d'attente dynamique permanente est supprimée, si le descripteur *HOBJ* indiqué dans l'appel MQCLOSE n'est pas celui qui a été renvoyé par l'appel MQOPEN qui a créé la file d'attente, il est vérifié que l'ID utilisateur utilisé pour valider l'appel MQOPEN est autorisé à supprimer la file d'attente. Si l'option OOALTU a été indiquée dans l'appel MQOPEN, l'ID utilisateur vérifié est *ODAU*.

Cette vérification n'est pas effectuée si:

- Le descripteur indiqué est celui renvoyé par l'appel MQOPEN qui a créé la file d'attente.
- La file d'attente en cours de suppression est une file d'attente dynamique temporaire.
- Lorsqu'une file d'attente dynamique temporaire est fermée, si le descripteur *HOBJ* indiqué dans l'appel MQCLOSE est celui qui a été renvoyé par l'appel MQOPEN qui a créé la file d'attente, la file d'attente est supprimée. Cela se produit quelles que soient les options de fermeture spécifiées dans l'appel MQCLOSE. S'il y a des messages dans la file d'attente, ils sont supprimés ; aucun message de rapport n'est généré.

S'il existe des unités de travail non validées qui affectent la file d'attente, la file d'attente et ses messages sont toujours supprimés, mais cela n'entraîne pas l'échec des unités de travail. Toutefois, comme décrit précédemment, les ressources associées aux unités de travail ne sont pas libérées tant que chacune des unités de travail n'a pas été validée ou annulée.

4. Les points suivants s'appliquent si l'objet en cours de fermeture est une *liste de distribution*:

- La seule option de fermeture valide pour une liste de distribution est CONONE ; l'appel échoue avec le code anomalie RC2046 ou RC2045 si d'autres options sont spécifiées.
- Lorsqu'une liste de distribution est fermée, les codes achèvement individuels et les codes raison ne sont pas renvoyés pour les files d'attente de la liste-seuls les paramètres **CMPCOD** et **REASON** de l'appel sont disponibles à des fins de diagnostic.

Si un incident se produit lors de la fermeture de l'une des files d'attente, le gestionnaire de files d'attente poursuit le traitement et tente de fermer les files d'attente restantes de la liste de distribution. Les paramètres **CMPCOD** et **REASON** de l'appel sont ensuite définis pour renvoyer des informations décrivant l'échec. Il est donc possible que le code achèvement soit CCFAIL, même si la plupart des files d'attente ont été fermées avec succès. La file d'attente qui a rencontré l'erreur n'est pas identifiée.

S'il y a un échec dans plusieurs files d'attente, il n'est pas défini quel échec est signalé dans les paramètres **CMPCOD** et **REASON**.

## Paramètres

L'appel MQCLOSE comporte les paramètres suivants:

### **HCONN (entier signé à 10 chiffres)-entrée**

Descripteur de connexion.

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *HCONN* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

### **HOBJ (entier signé à 10 chiffres)-entrée / sortie**

Descripteur d'objet.

Ce descripteur représente l'objet en cours de fermeture. L'objet peut être de n'importe quel type. La valeur de *HOBJ* a été renvoyée par un appel MQOPEN précédent.

Une fois l'appel terminé, le gestionnaire de files d'attente définit ce paramètre sur une valeur qui n'est pas un descripteur valide pour l'environnement. Cette valeur est:

#### **HOUNUH**

Descripteur d'objet inutilisable.

### **OPTS (entier signé à 10 chiffres)-entrée**

Options qui contrôlent l'action de MQCLOSE.

Le paramètre **OPTS** contrôle la manière dont l'objet est fermé. Seules les files d'attente dynamiques permanentes et les abonnements peuvent être fermés de plusieurs manières. Les files d'attente dynamiques permanentes peuvent être conservées ou supprimées ; il s'agit des files d'attente avec un attribut **DefinitionType** dont la valeur est QDPERM (voir l'attribut **DefinitionType** décrit dans

«Attributs des files d'attente», à la page 1435 ). Les options de fermeture sont récapitulées dans un tableau plus loin dans cette rubrique.

Les abonnements durables peuvent être conservés ou supprimés ; ils sont créés à l'aide de l'appel MQSUB avec l'option SODUR.

Lors de la fermeture de l'indicateur sur une destination gérée (c'est-à-dire le paramètre **Hobj** renvoyé sur un appel MQSUB qui utilisait l'option SOMAN), le gestionnaire de files d'attente nettoie toutes les publications non extraites lorsque l'abonnement associé a également été supprimé. Cette opération est effectuée à l'aide de l'option CORMSB sur le paramètre **Hsub** renvoyé lors d'un appel MQSUB. Notez que CORMSB est le comportement par défaut sur MQCLOSE pour un abonnement non durable.

Lors de la fermeture d'un descripteur vers une destination non gérée, vous êtes responsable du nettoyage de la file d'attente dans laquelle les publications sont envoyées. Il est recommandé de fermer l'abonnement à l'aide de CORMSB d'abord, puis de traiter les messages hors de la file d'attente jusqu'à ce qu'il n'en reste plus.

Un (et un seul) des éléments suivants doit être spécifié:

### Options de fermeture de file d'attente dynamique

Ces options contrôlent la manière dont les files d'attente dynamiques permanentes sont fermées:

#### MODÈLE

Supprimez la file d'attente.

La file d'attente est supprimée si l'une des conditions suivantes est vérifiée:

- Il s'agit d'une file d'attente dynamique permanente, créée par un précédent appel MQOPEN, et il n'y a aucun message dans la file d'attente et aucune demande d'extraction ou d'insertion non validée en attente pour la file d'attente (pour la tâche en cours ou toute autre tâche).
- Il s'agit de la file d'attente dynamique temporaire créée par l'appel MQOPEN qui a renvoyé **HOBJ**. Dans ce cas, tous les messages de la file d'attente sont purgés.

Dans tous les autres cas, y compris le cas où *Hobj* a été renvoyé sur un appel MQSUB, l'appel échoue avec le code anomalie RC2045et l'objet n'est pas supprimé.

#### COPURG

Supprimez la file d'attente, en purgeant les messages qu'elle contient.

La file d'attente est supprimée si l'une des conditions suivantes est vérifiée:

- Il s'agit d'une file d'attente dynamique permanente, créée par un appel MQOPEN précédent, et aucune demande d'extraction ou d'insertion non validée n'est en attente pour la file d'attente (pour la tâche en cours ou toute autre tâche).
- Il s'agit de la file d'attente dynamique temporaire créée par l'appel MQOPEN qui a renvoyé **HOBJ**.

Dans tous les autres cas, y compris le cas où *Hobj* a été renvoyé sur un appel MQSUB, l'appel échoue avec le code anomalie RC2045et l'objet n'est pas supprimé.

Le tableau suivant indique les options de fermeture valides et indique si l'objet est conservé ou supprimé.

Type d'objet ou de file d'attente	CONONE	MODÈLE	COPURG
Objet autre qu'une file d'attente	Conservé	Non valide	Non valide
File d'attente prédéfinie	Conservé	Non valide	Non valide

<i>Tableau 743. Options de fermeture valides à utiliser avec les objets conservés ou supprimés (suite)</i>			
<b>Type d'objet ou de file d'attente</b>	<b>CONONE</b>	<b>MODÈLE</b>	<b>COPURG</b>
file d'attente dynamique permanente	Conservé	Supprimé s'il est vide et qu'aucune mise à jour n'est en attente	Messages supprimés ; file d'attente supprimée si aucune mise à jour n'est en attente
File d'attente dynamique temporaire (appel émis par le créateur de la file d'attente)	Supprimé	Supprimé	Supprimé
File d'attente dynamique temporaire (appel non émis par le créateur de la file d'attente)	Conservé	Non valide	Non valide
Liste de distribution	Conservé	Non valide	Non valide
Destination d'abonnement géré	Conservé	Non valide	Non valide
Liste de distribution (l'abonnement a été supprimé)	Messages supprimés ; file d'attente supprimée	Non valide	Non valide

#### **Options de clôture de l'abonnement**

Ces options déterminent si les abonnements durables sont supprimés lorsque le descripteur est fermé et si les publications en attente de lecture par l'application sont nettoyées. Ces options ne peuvent être utilisées qu'avec un descripteur d'objet renvoyé dans le paramètre **HSUB** d'un appel MQSUB.

#### **COKPSB**

Le descripteur de l'abonnement est fermé, mais l'abonnement effectué est conservé. Les publications continueront d'être envoyées à la destination spécifiée dans l'abonnement. Cette option n'est valide que si l'abonnement a été effectué avec l'option SODUR. COKPSB est la valeur par défaut si l'abonnement est durable

#### **CORMSB**

L'abonnement est supprimé et le descripteur de l'abonnement est fermé.

Le paramètre **Hobj** de l'appel MQSUB n'est pas invalidé par la fermeture du paramètre **Hsub** et peut continuer à être utilisé pour que MQGET ou MQCB reçoive les publications restantes. Lorsque le paramètre **Hobj** de l'appel MQSUB est également fermé, s'il s'agissait d'une destination gérée, les publications non extraites sont supprimées.

CORMSB est la valeur par défaut si l'abonnement est non durable.

Ces options de clôture d'abonnement sont résumées dans les tableaux suivants:

Pour fermer un descripteur d'abonnement durable mais conserver l'abonnement, utilisez les options de fermeture d'abonnement suivantes:

<i>Tableau 744. Options de tâche pour la fermeture d'un descripteur d'abonnement durable et la fermeture de l'abonnement</i>	
<b>Tâche</b>	<b>Option de fermeture d'abonnement</b>
Conserver les publications sur un descripteur MQOPENed	COKPSB

Tableau 744. Options de tâche pour la fermeture d'un descripteur d'abonnement durable et la fermeture de l'abonnement (suite)

Tâche	Option de fermeture d'abonnement
Suppression de publications sur un descripteur MQOPENed	Action non autorisée
Conserver les publications sur une poignée avec SOMAN	COKPSB
Supprimer les publications sur un descripteur avec SOMAN	Action non autorisée

Pour vous désabonner, en fermant un descripteur d'abonnement durable et en le désabonnant ou en fermant un descripteur d'abonnement non durable, utilisez les options de fermeture d'abonnement suivantes:

Tableau 745. Options de tâche pour le désabonnement

Tâche	Option de fermeture d'abonnement
Conserver les publications sur un descripteur MQOPENed	CORMSB
Suppression de publications sur un descripteur MQOPENed	Action non autorisée
Conserver les publications sur une poignée avec SOMAN	CORMSB
Supprimer les publications sur un descripteur avec SOMAN	COPGSB

### Options de lecture anticipée

Les options suivantes contrôlent ce qui arrive aux messages non persistants qui ont été envoyés au client avant qu'une application ne les demande et qui n'ont pas encore été consommés par l'application. Ces messages sont stockés dans la mémoire tampon de lecture anticipée du client en attente d'être demandés par l'application et peuvent être supprimés ou consommés dans la file d'attente avant la fin de l'opération MQCLOSE.

#### COIMM

L'objet est fermé immédiatement et tous les messages qui ont été envoyés au client avant qu'une application ne les demande sont supprimés et ne peuvent être consommés par aucune application. Il s'agit de la valeur par défaut.

#### COQSC

Une demande de fermeture de l'objet est effectuée, mais si des messages qui ont été envoyés au client avant qu'une application ne les demande, résident toujours dans la mémoire tampon de lecture anticipée du client, l'appel MQCLOSE est renvoyé avec le code d'avertissement RC2458 et le descripteur d'objet reste valide.

L'application peut ensuite continuer à utiliser le descripteur d'objet pour extraire des messages jusqu'à ce qu'il ne soit plus disponible, puis fermer à nouveau l'objet. Aucun autre message ne sera envoyé au client avant une demande d'application, la lecture anticipée est maintenant désactivée.

Il est conseillé aux applications d'utiliser COQSC plutôt que d'essayer d'atteindre un point où il n'y a plus de messages dans la mémoire tampon de lecture anticipée du client, car un message pourrait arriver entre le dernier appel MQGET et le MQCLOSE suivant qui serait supprimé si COIMM était utilisé.

Si un MQCLOSE avec COQSC est émis à partir d'une fonction de rappel asynchrone, le même comportement de lecture des messages à l'avance s'applique. Si le code d'avertissement RC2458

est renvoyé, la fonction de rappel sera appelée au moins une fois de plus. Lorsque le dernier message restant ayant fait l'objet d'une lecture anticipée a été transmis à la fonction de rappel, la zone CBCFLG est définie sur CBCFBE.

### **Option par défaut**

Si vous n'avez besoin d'aucune des options décrites précédemment, vous pouvez utiliser l'option suivante:

#### **CONONE**

Aucun traitement de fermeture facultatif n'est requis.

Cette valeur doit être spécifiée pour:

- Objets autres que des files d'attente
- Files d'attente prédéfinies
- Files d'attente dynamiques temporaires (mais uniquement dans les cas où *HOBJ* n'est pas le descripteur renvoyé par l'appel *MQOPEN* qui a créé la file d'attente).
- Listes de diffusion

Dans tous les cas précédents, l'objet est conservé et non supprimé.

Si cette option est spécifiée pour une file d'attente dynamique temporaire:

- La file d'attente est supprimée si elle a été créée par l'appel *MQOPEN* qui a renvoyé *HOBJ* ; tous les messages qui se trouvent dans la file d'attente sont purgés.
- Dans tous les autres cas, la file d'attente (et tous les messages qu'elle contient) est conservée.

Si cette option est spécifiée pour une file d'attente dynamique permanente, la file d'attente est conservée et n'est pas supprimée.

### **CMPCOD (entier signé à 10 chiffres)-sortie**

Code achèvement.

Il peut s'agir de :

#### **CCOK**

Achèvement réussi.

#### **CCWARN**

Avertissement (achèvement partiel).

#### **CCFAIL**

Echec de l'appel.

### **REASON (entier signé à 10 chiffres)-sortie**

Code anomalie qualifiant *CMPCOD*.

Si *CMPCOD* est *CCOK*:

#### **RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CMPCOD* est *CCWARN*:

#### **RC2241**

(2241, X'8C1') Groupe de messages non complet.

#### **RC2242**

(2242, X'8C2') Message logique non complet.

Si *CMPCOD* est *CCFAIL*:

#### **RC2219**

(2219, X'8AB') Appel *MQI* entré à nouveau avant la fin de l'appel précédent.

#### **RC2009**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**RC2018**

(2018, X'7E2') Descripteur de connexion non valide.

**RC2019**

(2019, X'7E3') Descripteur d'objet non valide.

**RC2035**

(2035, X'7F3') Non autorisé pour l'accès.

**RC2101**

(2101, X'835') Objet endommagé.

**RC2045**

(2045, X'7FD') Option non valide pour le type d'objet.

**RC2046**

(2046, X'7FE') Options incorrectes ou incohérentes.

**RC2058**

(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

**RC2059**

(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

**RC2162**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**RC2055**

(2055, X'807') La file d'attente contient un ou plusieurs messages ou des demandes d'insertion ou d'extraction non validées.

**RC2102**

(2102, X'836') Ressources système disponibles insuffisantes.

**RC2063**

(2063, X'80F') Une erreur de sécurité s'est produite.

**RC2071**

(2071, X'817') Mémoire disponible insuffisante.

**RC2195**

(2195, X'893') Une erreur inattendue s'est produite.

## Déclaration RPG

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQCLOSE(HCONN : HOBJ : OPTS :
C                               CMPCOD : REASON)

```

La définition de prototype pour l'appel est la suivante:

```

D*..1.....2.....3.....4.....5.....6.....7..
DMQCLOSE      PR          EXTPROC('MQCLOSE')
D* Connection handle
D HCONN              10I 0 VALUE
D* Object handle
D HOBJ              10I 0
D* Options that control the action of MQCLOSE
D OPTS              10I 0 VALUE
D* Completion code
D CMPCOD            10I 0
D* Reason code qualifying CMPCOD
D REASON            10I 0

```

## MQCMIT (Valider les modifications) sous IBM i

L'appel MQCMIT indique au gestionnaire de files d'attente que l'application a atteint un point de synchronisation et que toutes les opérations d'obtention et d'insertion de messages qui se sont produites



depuis le dernier point de synchronisation doivent être rendues permanentes. Les messages insérés dans une unité de travail sont mis à la disposition d'autres applications ; les messages extraits dans le cadre d'une unité de travail sont supprimés.

- [«Syntaxe», à la page 1337](#)
- [«Remarques sur l'utilisation», à la page 1337](#)
- [«Paramètres», à la page 1338](#)
- [«Déclaration RPG», à la page 1339](#)

## Syntaxe

MQCMIT (*HCONN, COMCOD, REASON*)

## Remarques sur l'utilisation

Tenez compte de ces remarques d'utilisation lorsque vous utilisez MQCMIT.

1. Cet appel peut être utilisé uniquement lorsque le gestionnaire de files d'attente lui-même coordonne l'unité d'oeuvre. Il s'agit d'une unité d'oeuvre locale, où les modifications affectent uniquement les ressources IBM MQ .
2. Dans les environnements où le gestionnaire de files d'attente ne coordonne pas l'unité d'oeuvre, l'appel de validation approprié doit être utilisé à la place de MQCMIT. L'environnement peut également prendre en charge une validation implicite provoquée par l'arrêt normal de l'application.
  - Sous IBM i, cet appel peut être utilisé pour les unités d'oeuvre locales coordonnées par le gestionnaire de files d'attente. Cela signifie qu'une définition de validation ne doit pas exister au niveau du travail, c'est-à-dire que la commande STRCMTCTL avec le paramètre **CMTSCOPE (\*JOB)** ne doit pas avoir été émise pour le travail.
3. Si une application se termine avec des modifications non validées dans une unité d'oeuvre, la disposition de ces modifications varie selon que l'application se termine normalement ou anormalement. Pour plus de détails, voir les remarques sur l'utilisation dans [«MQDISC \(gestionnaire de files d'attente de déconnexion\) sous IBM i», à la page 1353](#) .
4. Lorsqu'une application insère ou extrait des messages dans des groupes ou des segments de messages logiques, le gestionnaire de files d'attente conserve les informations relatives au groupe de messages et au message logique pour les derniers appels MQPUT et MQGET ayant abouti. Ces informations sont associées à l'identificateur de file d'attente et incluent les éléments suivants:
  - Valeurs des zones *MDGID, MDSEQ, MDOFF* et *MDMFL* dans MQMD.
  - Indique si le message fait partie d'une unité de travail.
  - Pour l'appel MQPUT: indique si le message est persistant ou non persistant.

Lorsqu'une unité d'oeuvre est validée, le gestionnaire de files d'attente conserve les informations de groupe et de segment et l'application peut continuer à insérer ou à extraire des messages dans le groupe de messages ou le message logique en cours.

La conservation des informations de groupe et de segment lors de la validation d'une unité de travail permet à l'application de répartir un grand groupe de messages ou un grand message logique composé de plusieurs segments sur plusieurs unités de travail. L'utilisation de plusieurs unités d'oeuvre peut s'avérer avantageuse si le gestionnaire de files d'attente local ne dispose que d'une mémoire de file d'attente limitée. Toutefois, l'application doit conserver suffisamment d'informations pour pouvoir redémarrer l'insertion ou l'obtention de messages au point approprié en cas de défaillance du système. Pour plus d'informations sur le redémarrage au point approprié après une défaillance du système, voir l'option PMLOGO décrite dans [«MQPMO \(options d'insertion de message\) sous IBM i», à la page 1230](#) et l'option GMLOGO décrite dans [«MQGMO \(options d'extraction de message\) sous IBM i», à la page 1127](#).

Les autres remarques d'utilisation s'appliquent uniquement lorsque le gestionnaire de files d'attente coordonne les unités de travail:

1. Une unité de travail a la même portée qu'un descripteur de connexion. Cela signifie que tous les appels IBM MQ qui affectent une unité de travail particulière doivent être exécutés à l'aide du même descripteur de connexion. Les appels émis à l'aide d'un descripteur de connexion différent (par exemple, les appels émis par une autre application) affectent une unité d'oeuvre différente. Pour plus d'informations sur la portée des descripteurs de connexion, voir le paramètre **HCONN** décrit dans MQCONN.
2. Seuls les messages insérés ou extraits dans le cadre de l'unité d'oeuvre en cours sont affectés par cet appel.
3. Une application à exécution longue qui émet des appels MQGET, MQPUT ou MQPUT1 dans une unité de travail, mais qui n'émet jamais d'appel de validation ou d'annulations, peut provoquer le remplissage des files d'attente avec des messages qui ne sont pas disponibles pour d'autres applications. Pour éviter cette possibilité, l'administrateur doit définir l'attribut de gestionnaire de files d'attente **MaxUncommittedMsgs** sur une valeur suffisamment faible pour empêcher les applications en fuite de remplir les files d'attente, mais suffisamment élevée pour permettre aux applications de messagerie attendues de fonctionner correctement.

## Paramètres

L'appel MQCMIT comporte les paramètres suivants:

### **HCONN (entier signé à 10 chiffres)-entrée**

Descripteur de connexion.

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *HCONN* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

### **COMCOD (entier signé à 10 chiffres)-sortie**

Code achèvement.

Il peut s'agir de :

#### **CCOK**

Achèvement réussi.

#### **CCWARN**

Avertissement (achèvement partiel).

#### **CCFAIL**

Echec de l'appel.

### **REASON (entier signé à 10 chiffres)-sortie**

Code anomalie qualifiant *COMCOD*.

Si *COMCOD* est CCOK:

#### **RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si *COMCOD* est CCWARN:

#### **RC2003**

(2003, X'7D3') Unité d'oeuvre annulée.

#### **RC2124**

(2124, X'84C') Le résultat de l'opération de validation est en attente.

Si *COMCOD* est CCFAIL:

#### **RC2219**

(2219, X'8AB') Appel MQI entré à nouveau avant la fin de l'appel précédent.

**RC2009**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**RC2018**

(2018, X'7E2') Descripteur de connexion non valide.

**RC2101**

(2101, X'835') Objet endommagé.

**RC2123**

(2123, X'84B') Le résultat de l'opération de validation ou d'exclusion est mélangé.

**RC2162**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**RC2102**

(2102, X'836') Ressources système disponibles insuffisantes.

**RC2071**

(2071, X'817') Mémoire disponible insuffisante.

**RC2195**

(2195, X'893') Une erreur inattendue s'est produite.

**Déclaration RPG**

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP          MQCMIT(HCONN : COMCOD : REASON)

```

La définition de prototype pour l'appel est la suivante:

```

D*..1.....2.....3.....4.....5.....6.....7..
DMQCMIT          PR          EXTPROC('MQCMIT')
D* Connection handle
D HCONN          10I 0 VALUE
D* Completion code
D COMCOD          10I 0
D* Reason code qualifying COMCOD
D REASON          10I 0

```

**IBM i MQCONN (Connect queue manager) sur IBM i**

L'appel MQCONN connecte un programme d'application à un gestionnaire de files d'attente. Il fournit un descripteur de connexion de gestionnaire de files d'attente, qui est utilisé par l'application lors des appels de mise en file d'attente de messages ultérieurs.

- Les applications doivent utiliser l'appel MQCONN ou MQCONNX pour se connecter au gestionnaire de files d'attente et l'appel MQDISC pour se déconnecter du gestionnaire de files d'attente.

Sous IBM MQ for Multiplatforms, chaque unité d'exécution d'une application peut se connecter à différents gestionnaires de files d'attente. Sur les autres systèmes, toutes les connexions simultanées d'un processus doivent être établies avec le même gestionnaire de files d'attente.

- «Syntaxe», à la page [1339](#)
- «Remarques sur l'utilisation», à la page [1340](#)
- «Paramètres», à la page [1340](#)
- «Déclaration RPG», à la page [1343](#)

**Syntaxe**

MQCONN (QMNAME, HCONN, CMPCOD, REASON)

## Remarques sur l'utilisation

1. Le gestionnaire de files d'attente auquel la connexion est établie à l'aide de l'appel MQCONN est appelé *gestionnaire de files d'attente local*.
2. Les files d'attente appartenant au gestionnaire de files d'attente local apparaissent à l'application en tant que files d'attente locales. Il est possible d'insérer et d'extraire des messages de ces files d'attente.

Les files d'attente partagées appartenant au groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local apparaissent dans l'application en tant que files d'attente locales. Il est possible d'insérer et d'extraire des messages de ces files d'attente.

Les files d'attente appartenant à des gestionnaires de files d'attente éloignées apparaissent comme des files d'attente éloignées. Il est possible d'insérer des messages dans ces files d'attente, mais il n'est pas possible d'en extraire.

3. Si le gestionnaire de files d'attente échoue alors qu'une application est en cours d'exécution, l'application doit émettre à nouveau l'appel MQCONN afin d'obtenir un nouveau descripteur de connexion à utiliser lors des appels IBM MQ suivants. L'application peut émettre l'appel MQCONN périodiquement jusqu'à ce que l'appel aboutisse.

Si une application ne sait pas si elle est connectée au gestionnaire de files d'attente, elle peut émettre un appel MQCONN en toute sécurité afin d'obtenir un descripteur de connexion. Si l'application est déjà connectée, le descripteur renvoyé est identique à celui renvoyé par l'appel MQCONN précédent, mais avec le code achèvement CCWARN et le code anomalie RC2002.

4. Lorsque l'application a fini d'utiliser les appels IBM MQ, elle doit utiliser l'appel MQDISC pour se déconnecter du gestionnaire de files d'attente.
5. Sous IBM i, les programmes qui se terminent de manière anormale ne sont pas automatiquement déconnectés du gestionnaire de files d'attente. Par conséquent, les applications doivent être écrites pour permettre la possibilité que l'appel MQCONN ou MQCONNX renvoie le code achèvement CCWARN et le code anomalie RC2002. Le descripteur de connexion renvoyé dans cette situation peut être utilisé normalement.

## Paramètres

L'appel MQCONN comporte les paramètres suivants:

### QMNAME (chaîne de caractères de 48 octets)-entrée

Nom du gestionnaire de files d'attente.

Il s'agit du nom du gestionnaire de files d'attente auquel l'application souhaite se connecter. Le nom peut contenir les caractères suivants:

- Caractères alphabétiques en majuscules (A à Z)
- Caractères alphabétiques minuscules (a à z)
- Chiffres (0 à 9)
- Point (.), barre oblique (/), trait de soulignement (\_), pourcentage (%)

Le nom ne doit pas contenir de blancs de début ou imbriqués, mais peut contenir des blancs de fin. Un caractère NULL peut être utilisé pour indiquer la fin des données significatives dans le nom ; la valeur NULL et les caractères qui la suivent sont traités comme des blancs. Les restrictions suivantes s'appliquent dans les environnements indiqués:

- Sous IBM i, les noms contenant des caractères minuscules, des barres obliques ou des pourcentages doivent être placés entre guillemets lorsqu'ils sont spécifiés dans les commandes. Ces guillemets ne doivent pas être spécifiés dans le paramètre **QMNAME**.

Si le nom est entièrement vide, le nom du gestionnaire de files d'attente *par défaut* est utilisé.

Le nom spécifié pour **QMNAME** doit être le nom d'un gestionnaire de files d'attente *connectable*.

**Groupes de partage de files d'attente:** Sur les systèmes où plusieurs gestionnaires de files d'attente existent et sont configurés pour former un groupe de partage de files d'attente, le nom du groupe de partage de files d'attente peut être spécifié pour *QMNAME* à la place du nom d'un gestionnaire de files d'attente. Cela permet à l'application de se connecter à *tout gestionnaire de files d'attente* disponible dans le groupe de partage de files d'attente. Le système peut également être configuré de sorte qu'un *QMNAME* vide entraîne la connexion au groupe de partage de files d'attente au lieu du gestionnaire de files d'attente par défaut.

Si *QMNAME* spécifie le nom du groupe de partage de files d'attente, mais qu'il existe également un gestionnaire de files d'attente portant ce nom sur le système, la connexion est établie à ce dernier de préférence au premier. La connexion à l'un des gestionnaires de files d'attente du groupe de partage de files d'attente n'est tentée que si cette connexion échoue.

Si la connexion aboutit, l'identificateur renvoyé par l'appel MQCONN ou MQCONNX peut être utilisé pour accéder à *toutes* les ressources (partagées et non partagées) appartenant au gestionnaire de files d'attente particulier auquel la connexion a été établie. L'accès à ces ressources est soumis aux contrôles d'autorisation standard.

Si l'application émet deux appels MQCONN ou MQCONNX afin d'établir des connexions simultanées et qu'un ou les deux appels indiquent le nom du groupe de partage de files d'attente, le deuxième appel peut renvoyer le code d'achèvement CCWARN et le code anomalie RC2002. Cela se produit lorsque le deuxième appel se connecte au même gestionnaire de files d'attente que le premier appel.

Les groupes de partage de files d'attente sont pris en charge uniquement sous z/OS. La connexion à un groupe de partage de files d'attente n'est prise en charge que dans les environnements par lots, RRS par lots et TSO.

**Applications client IBM MQ:** pour les applications IBM MQ MQI client, une connexion est tentée pour chaque définition de canal de connexion client avec le nom de gestionnaire de files d'attente spécifié, jusqu'à ce qu'une connexion aboutisse. Toutefois, le gestionnaire de files d'attente doit avoir le même nom que le nom spécifié. Si un nom vide est spécifié, chaque canal de connexion client avec un nom de gestionnaire de files d'attente vide est essayé jusqu'à ce qu'il aboutisse ; dans ce cas, il n'y a pas de vérification par rapport au nom réel du gestionnaire de files d'attente.

**Groupes de gestionnaires de files d'attente client IBM MQ:** si le nom spécifié commence par un astérisque (\*), le nom du gestionnaire de files d'attente réel auquel la connexion est établie peut être différent de celui spécifié par l'application. Le nom spécifié (sans astérisque) définit un *groupe* de gestionnaires de files d'attente éligibles pour la connexion. L'implémentation en sélectionne un dans le groupe en essayant chacun à son tour, dans l'ordre alphabétique, jusqu'à ce qu'une connexion soit trouvée. Si aucun des gestionnaires de files d'attente du groupe n'est disponible pour la connexion, l'appel échoue. Chaque gestionnaire de files d'attente est essayé une seule fois. Si un astérisque seul est spécifié pour le nom, un groupe de gestionnaires de files d'attente par défaut défini par l'implémentation est utilisé.

Les groupes de gestionnaires de files d'attente sont pris en charge uniquement pour les applications s'exécutant dans un environnement MQ-client ; l'appel échoue si une application non client spécifie un nom de gestionnaire de files d'attente commençant par un astérisque. Un groupe est défini en fournissant plusieurs définitions de canal de connexion client avec le même nom de gestionnaire de files d'attente (nom spécifié sans astérisque), pour communiquer avec chacun des gestionnaires de files d'attente du groupe. Le groupe par défaut est défini en fournissant une ou plusieurs définitions de canal de connexion client, chacune avec un nom de gestionnaire de files d'attente vide (la spécification d'un nom entièrement vide a donc le même effet que la spécification d'un astérisque unique pour le nom d'une application client).

Après la connexion à un gestionnaire de files d'attente d'un groupe, une application peut spécifier des blancs de la manière habituelle dans les zones de nom de gestionnaire de files d'attente dans les descripteurs de message et d'objet pour désigner le nom du gestionnaire de files d'attente auquel l'application s'est connectée (le *gestionnaire de files d'attente local*). Si l'application a besoin de connaître ce nom, l'appel MQINQ peut être émis pour interroger l'attribut de gestionnaire de files d'attente **QMGrName**.

Le fait d'ajouter un astérisque au nom de connexion implique que l'application ne dépend pas de la connexion à un gestionnaire de files d'attente particulier du groupe. Les applications appropriées seraient les suivantes:

- Applications qui placent des messages mais n'en reçoivent pas.
- Applications qui placent des messages de demande, puis extraient les messages de réponse d'une file d'attente *dynamique temporaire*.

Les applications qui ne conviennent pas sont celles qui ont besoin d'obtenir des messages d'une file d'attente particulière sur un gestionnaire de files d'attente particulier ; ces applications ne doivent pas préfixer le nom avec un astérisque.

Notez que si un astérisque est indiqué, la longueur maximale du reste du nom est de 47 caractères.

La longueur de ce paramètre est indiquée par LNQMN.

### **HCONN (entier signé à 10 chiffres)-sortie**

Descripteur de connexion.

Ce descripteur représente la connexion au gestionnaire de files d'attente. Il doit être spécifié sur tous les appels de mise en file d'attente de messages ultérieurs émis par l'application. Elle cesse d'être valide lorsque l'appel MQDISC est émis ou lorsque l'unité de traitement qui définit la portée de l'identificateur s'arrête.

La portée de la poignée est limitée à la plus petite unité de traitement parallèle pris en charge par la plateforme sur laquelle l'application s'exécute ; le descripteur n'est pas valide en dehors de l'unité de traitement parallèle à partir de laquelle l'appel MQCONN a été émis.

- Sous IBM i, la portée du descripteur est le travail qui émet l'appel.

### **CMPCOD (entier signé à 10 chiffres)-sortie**

Code achèvement.

Il peut s'agir de :

#### **CCOK**

Achèvement réussi.

#### **CCWARN**

Avertissement (achèvement partiel).

#### **CCFAIL**

Echec de l'appel.

### **REASON (entier signé à 10 chiffres)-sortie**

Code anomalie qualifiant *CMPCOD*.

Si *CMPCOD* est CCOK:

#### **RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CMPCOD* est CCWARN:

#### **RC2002**

(2002, X'7D2') Application déjà connectée.

Si *CMPCOD* est CCFAIL:

#### **RC2219**

(2219, X'8AB') Appel MQI entré à nouveau avant la fin de l'appel précédent.

#### **RC2267**

(2267, X'8DB') Impossible de charger l'exit de charge de travail du cluster.

#### **RC2009**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**RC2018**

(2018, X'7E2') Descripteur de connexion non valide.

**RC2035**

(2035, X'7F3') Non autorisé pour l'accès.

**RC2137**

(2137, X'859') L'ouverture de l'objet a échoué.

**RC2058**

(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

**RC2059**

(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

**RC2161**

(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.

**RC2162**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**RC2102**

(2102, X'836') Ressources système disponibles insuffisantes.

**RC2063**

(2063, X'80F') Une erreur de sécurité s'est produite.

**RC2071**

(2071, X'817') Mémoire disponible insuffisante.

**RC2195**

(2195, X'893') Une erreur inattendue s'est produite.

**Déclaration RPG**

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQCONN(QMNAME : HCONN : CMPCOD :
C                               REASON)

```

La définition de prototype pour l'appel est la suivante:

```

D*..1.....2.....3.....4.....5.....6.....7..
DMQCONN      PR          EXTPROC('MQCONN')
D* Name of queue manager
D QMNAME          48A
D* Connection handle
D HCONN          10I 0
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CMPCOD
D REASON          10I 0

```

## **MQCONNX (gestionnaire de files d'attente de connexion (étendu))** sous IBM i

L'appel MQCONNX connecte un programme d'application à un gestionnaire de files d'attente. Il fournit un descripteur de connexion de gestionnaire de files d'attente, qui est utilisé par l'application lors des appels IBM MQ suivants.

L'appel MQCONNX est similaire à l'appel MQCONN, sauf que MQCONNX permet de spécifier des options pour contrôler la façon dont l'appel fonctionne.

Sous IBM MQ for Multiplatforms, chaque unité d'exécution d'une application peut se connecter à différents gestionnaires de files d'attente. Sur les autres systèmes, toutes les connexions simultanées d'un processus doivent être établies avec le même gestionnaire de files d'attente.

- «Syntaxe», à la page 1344
- «Paramètres», à la page 1344
- «Déclaration RPG», à la page 1344

## Syntaxe

(QMNAME, CNOPT, HCONN, CMPCOD, REASON) MQCONNX

## Paramètres

L'appel MQCONNX comporte les paramètres suivants:

### QMNAME (chaîne de caractères de 48 octets)-entrée

Nom du gestionnaire de files d'attente.

Pour plus de détails, voir le paramètre **QMNAME** décrit dans [«MQCONN \(Connect queue manager\) sur IBM i», à la page 1339](#).

### CNOPT (MQCNO)-entrée/sortie

Options qui contrôlent l'action de MQCONNX.

Pour plus d'informations, voir [«MQCNO \(options de connexion\) sous IBM i», à la page 1097](#).

### HCONN (entier signé à 10 chiffres)-sortie

Descripteur de connexion.

Pour plus de détails, voir le paramètre **HCONN** décrit dans [«MQCONN \(Connect queue manager\) sur IBM i», à la page 1339](#).

### CMPCOD (entier signé à 10 chiffres)-sortie

Code achèvement.

Pour plus de détails, voir le paramètre **CMPCOD** décrit dans [«MQCONN \(Connect queue manager\) sur IBM i», à la page 1339](#).

### REASON (entier signé à 10 chiffres)-sortie

Code anomalie qualifiant *CMPCOD*.

Pour plus de détails sur les codes anomalie possibles, voir le paramètre **REASON** décrit dans [«MQCONN \(Connect queue manager\) sur IBM i», à la page 1339](#).

Les codes anomalie supplémentaires suivants peuvent être renvoyés par l'appel MQCONNX:

Si *CMPCOD* est CCFAIL:

#### RC2278

(2278, X'8E6') Zones de connexion client non valides.

#### RC2139

(2139, X'85B') Structure d'options de connexion non valide.

#### RC2046

(2046, X'7FE') Options incorrectes ou incohérentes.

## Déclaration RPG

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQCONN(QMNAME : HCONN : CMPCOD :
C                               REASON)

```

La définition de prototype pour l'appel est la suivante:



```

D*..1.....2.....3.....4.....5.....6.....7..
DMQCONN          PR          EXTPROC('MQCONN')
D* Name of queue manager
D QMNAME          48A
D* Options that control the action of MQCONN
D HCONN          224A
D* Connection handle
D HCONN          10I 0
D* Completion code
D CMPCOD         10I 0
D* Reason code qualifying CMPCOD
D REASON         10I 0

```

## IBM i MQCRTMH (Create message handle) sur IBM i

L'appel MQCRTMH renvoie un descripteur de message.

Une application peut l'utiliser lors des appels de mise en file d'attente de messages suivants:

- Utilisez l'appel [MQSETMP](#) pour définir une propriété du descripteur de message.
- Utilisez l'appel [MQINQMP](#) pour demander la valeur d'une propriété du descripteur de message.
- Utilisez l'appel [MQDLTMP](#) pour supprimer une propriété du descripteur de message.

Le descripteur de message peut être utilisé sur les appels MQPUT et MQPUT1 pour associer les propriétés du descripteur de message aux propriétés du message en cours d'insertion. De même, en spécifiant un descripteur de message sur l'appel MQGET, les propriétés du message en cours d'extraction sont accessibles à l'aide du descripteur de message lorsque l'appel MQGET est terminé.

Utilisez [MQDLTMH](#) pour supprimer le descripteur de message.

- «Syntaxe», à la page [1345](#)
- «Paramètres», à la page [1345](#)
- «Déclaration RPG», à la page [1347](#)

### Syntaxe

(*Hconn*, *CrtMsgHOpts*, *Hmsg*, *CompCode*, *Reason*) MQCRTMH

### Paramètres

L'appel MQCRTMH comporte les paramètres suivants:

#### HCONN (entier signé à 10 chiffres)-entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *HCONN* a été renvoyée par un appel MQCONN ou MQCONNX précédent. Si la connexion au gestionnaire de files d'attente cesse d'être valide et qu'aucun appel IBM MQ n'est en cours d'exécution sur l'identificateur de message, [MQDLTMH](#) est implicitement appelé pour supprimer le message.

Vous pouvez également spécifier la valeur suivante:

#### HCUNAS

Le descripteur de connexion ne représente pas une connexion à un gestionnaire de files d'attente particulier.

Lorsque cette valeur est utilisée, l'identificateur de message doit être supprimé avec un appel explicite à [MQDLTMH](#) afin de libérer la mémoire qui lui est allouée ; IBM MQ ne supprime jamais implicitement l'identificateur de message.

Il doit y avoir au moins une connexion valide à un gestionnaire de files d'attente établi sur l'unité d'exécution qui crée le descripteur de message, sinon l'appel échoue avec RC2018.

#### CROPT (MQCMHO)-entrée

Options qui contrôlent l'action de MQCRTMH. Pour plus d'informations, voir [MQCMHO](#) .

## **HMSG (entier signé à 20 chiffres)-sortie**

En sortie, un descripteur de message est renvoyé et peut être utilisé pour définir, interroger et supprimer les propriétés du descripteur de message. Initialement, le descripteur de message ne contient aucune propriété.

Un descripteur de message est également associé à un descripteur de message. Initialement, ce descripteur de message contient les valeurs par défaut. Les valeurs des zones de descripteur de message associées peuvent être définies et renseignées à l'aide des appels MQSETMP et MQINQMP. L'appel MQDLTMP réinitialise une zone du descripteur de message à sa valeur par défaut.

Si le paramètre *HCONN* est spécifié en tant que valeur HCUNAS, le descripteur de message renvoyé peut être utilisé sur les appels MQGET, MQPUT ou MQPUT1 avec n'importe quelle connexion dans l'unité de traitement, mais il ne peut être utilisé que par un seul appel IBM MQ à la fois. Si le descripteur est utilisé lorsqu'un deuxième appel IBM MQ tente d'utiliser le même descripteur de message, le deuxième appel IBM MQ échoue avec le code anomalie RC2499.

Si le paramètre *HCONN* n'est pas HCUNAS, le descripteur de message renvoyé ne peut être utilisé que sur la connexion spécifiée.

La même valeur de paramètre *HCONN* doit être utilisée sur les appels MQI suivants où ce descripteur de message est utilisé:

- MQDLTMH
- MQSETMP
- MQINQMP
- MQDLTMP
- MQMHBUF
- MQBUFMH

L'identificateur de message renvoyé cesse d'être valide lorsque l'appel MQDLTMH est émis pour l'identificateur de message ou lorsque l'unité de traitement qui définit la portée de l'identificateur s'arrête. MQDLTMH est appelé implicitement si une connexion spécifique est fournie lors de la création du descripteur de message et que la connexion au gestionnaire de files d'attente cesse d'être valide, par exemple, si MQDBC est appelé.

## **CMPCOD (entier signé à 10 chiffres)-sortie**

Code achèvement ; il s'agit d'un des codes suivants :

### **CCOK**

Achèvement réussi.

### **CCFAIL**

Echec de l'appel.

## **REASON (entier signé à 10 chiffres)-sortie**

Code anomalie qualifiant *CMPCOD*.

Si *CMPCOD* est CCOK:

### **RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CMPCOD* est CCFAIL:

### **RC2204**

(2204, X'089C') Carte non disponible.

### **RC2130**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

### **RC2157**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

**RC2219**

(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

**RC2461**

(2461, X'099D') La structure des options de création de descripteur de message n'est pas valide.

**RC2273**

(2273, X'7D9') La connexion au gestionnaire de files d'attente a été perdue.

**RC2017**

(2017, X'07E1') Plus de descripteurs disponibles.

**RC2018**

(2018, X'7E2') Descripteur de connexion non valide.

**RC2460**

(2460, X'099C') Le pointeur de descripteur de message n'est pas valide.

**RC2046**

(2046, X'07FE') Options non valides ou non cohérentes.

**RC2071**

(2071, X'817') Mémoire disponible insuffisante.

**RC2195**

(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations, voir [«Codes retour pour IBM i \(ILE RPG\)»](#), à la page 1498.

**Déclaration RPG**

```

C*.1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQCRTMH(HCONN : CRTOPT : HMSG :
                          CMPCOD : REASON)

```

La définition de prototype pour l'appel est la suivante:

```

DMQCRTMH          PR          EXTPROC('MQCRTMH')
D* Connection handle
D HCONN          10I 0 VALUE
D* Options that control the action of MQCRTMH
D CRTOPT          12A
D* Message handle
D HMSG          20I 0
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CompCode
D REASON          10I 0

```

**IBM i MQCTL (rappel de contrôle) sous IBM i**

L'appel MQCTL effectue des actions de contrôle sur les descripteurs d'objet ouverts pour une connexion.

- [«Syntaxe»](#), à la page 1347
- [«Remarques sur l'utilisation»](#), à la page 1348
- [«Paramètres»](#), à la page 1348
- [«Déclaration RPG»](#), à la page 1353

**Syntaxe**

MQCTL (*Hconn, Operation, ControlOpts, CompCode, Reason*)

## Remarques sur l'utilisation

1. Les routines de rappel doivent vérifier les réponses de tous les services qu'elles appellent et si elles détectent une condition qui ne peut pas être résolue, elles doivent émettre une commande MQCB (CBREG) pour empêcher les appels répétés à la routine de rappel.

## Paramètres

L'appel MQCTL comporte les paramètres suivants:

### HCONN (entier signé à 10 chiffres)-entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *HCONN* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

### OPERATN (entier signé à 10 chiffres)-entrée

Opération en cours de traitement sur le rappel défini pour le descripteur d'objet spécifié. Vous devez spécifier une seule des options suivantes:

#### CTLSR

Démarrer la consommation des messages pour toutes les fonctions de consommateur de message définies pour le descripteur de connexion spécifié.

Les rappels s'exécutent sur une unité d'exécution démarrée par le système, qui est différente de toutes les unités d'exécution d'application.

Cette opération permet de contrôler la poignée de connexion fournie au système. Les seuls appels MQI pouvant être émis par une unité d'exécution autre que l'unité d'exécution destinataire sont les suivants:

- MQCTL avec l'opération CTLSP
- MQCTL avec l'opération CTLSU
- MQDISC-Permet d'exécuter MQCTL avec l'opération CTLSP avant de déconnecter HConn.

RC2500 est renvoyé si un appel d'API IBM MQ est émis alors que le descripteur de connexion est démarré et que l'appel ne provient pas d'une fonction de consommateur de message.

Si une connexion échoue, la conversation est arrêtée dès que possible. Il est donc possible qu'un appel d'API IBM MQ soit émis sur l'unité d'exécution principale pour recevoir le code retour RC2500 pendant un certain temps, suivi du code retour RC2009 lorsque la connexion repasse à l'état arrêté.

Cela peut être émis dans une fonction de consommateur. Pour la même connexion que la routine de rappel, son seul objectif est d'annuler une opération CTLSP précédemment émise.

Cette option n'est pas prise en charge si l'application est liée à une bibliothèque IBM MQ non à unités d'exécution.

#### CTLSW

Démarrer la consommation des messages pour toutes les fonctions de consommateur de message définies pour le descripteur de connexion spécifié.

Les consommateurs de message s'exécutent sur la même unité d'exécution et le contrôle n'est pas renvoyé à l'appelant de MQCTL tant que:

- Publié par l'utilisation des opérations MQCTL CTLSP ou CTLSU, ou
- Toutes les routines de consommateur ont été désenregistrées ou suspendues.

Si tous les consommateurs sont désenregistrés ou suspendus, une opération CTLSP implicite est émise.

Cette option ne peut pas être utilisée à partir d'une routine de rappel, que ce soit pour le descripteur de connexion en cours ou tout autre descripteur de connexion. Si l'appel est tenté, il est renvoyé avec RC2012.

Si, à tout moment au cours d'une opération CTLSW, il n'y a pas de consommateurs enregistrés et non suspendus, l'appel échoue avec le code anomalie RC2446.

Si, lors d'une opération CTLSW, la connexion est interrompue, l'appel MQCTL renvoie le code anomalie d'avertissement RC2521; la connexion reste 'démarrée'.

L'application peut choisir d'émettre des CTLSP ou des CTLRE. Dans ce cas, l'opération CTLRE se bloque.

Cette option n'est pas prise en charge dans un client à unité d'exécution unique.

### **CTLSP**

Arrêtez la consommation des messages et attendez que tous les destinataires aient terminé leurs opérations avant de terminer cette option. Cette opération libère le descripteur de connexion.

Si elle est émise à partir d'une routine de rappel, cette option n'est pas appliquée tant que la routine n'est pas arrêtée. Aucune autre routine de consommateur de message n'est appelée une fois que les routines de consommateur des messages déjà lus sont terminées et après que des appels d'arrêt (si requis) ont été effectués aux routines de rappel.

S'il est émis en dehors d'une routine de rappel, le contrôle ne revient pas à l'appelant tant que les routines de consommateur pour les messages déjà lus ne sont pas terminées et que des appels d'arrêt (si requis) aux rappels n'ont pas été effectués. Les rappels eux-mêmes, cependant, restent enregistrés.

Cette fonction n'a aucun effet sur les messages de lecture anticipée. Vous devez vous assurer que les consommateurs exécutent MQCLOSE (COQSC), à partir de la fonction de rappel, pour déterminer si d'autres messages sont disponibles pour être distribués.

### **CTLSU**

Mettez en pause la consommation des messages. Cette opération libère le descripteur de connexion.

Cela n'affecte pas la lecture avant les messages de l'application. Si vous avez l'intention d'arrêter de consommer des messages pendant une longue période, envisagez de fermer la file d'attente et de la rouvrir lorsque la consommation doit continuer.

S'il est émis à partir d'une routine de rappel, il ne prend effet qu'à la fin de la routine. Aucune autre routine de consommateur de message ne sera appelée après la sortie de la routine en cours.

S'il est émis en dehors d'un rappel, le contrôle ne revient pas à l'appelant tant que la routine de consommateur en cours n'est pas terminée et qu'aucun autre n'est appelé.

### **CTLRE**

Reprenez la consommation des messages.

Cette option est normalement émise à partir de l'unité d'exécution de l'application principale, mais elle peut également être utilisée à partir d'une routine de rappel pour annuler une demande de suspension antérieure émise dans la même routine.

Si CTLRE est utilisé pour reprendre une CTLSW, l'opération se bloque.

### **PCTLOP (MQCTLO)-entrée**

Options qui contrôlent l'action de MQCTL

Pour plus d'informations sur la structure, voir [MQCTLO](#) .

### **CMPCOD (entier signé à 10 chiffres)-sortie**

Code achèvement ; il s'agit d'un des codes suivants :

#### **CCOK**

Achèvement réussi.

#### **CCWARN**

Avertissement (achèvement partiel).

**CCFAIL**

Echec de l'appel.

**REASON (entier signé à 10 chiffres)-sortie**

Les codes anomalie suivants sont ceux que le gestionnaire de files d'attente peut renvoyer pour le paramètre **Reason** .

Si *CMPCOD* est CCOK:

**RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CMPCOD* est CCFAIL:

**RC2133**

(2133, X'855') Impossible de charger les modules de service de conversion de données.

**RC2204**

(2204, X'89C') Adaptateur non disponible.

**RC2130**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

**RC2374**

(2374, X'946') Echec de l'exit API.

**RC2183**

(2183, X'887') Impossible de charger l'exit API.

**RC2157**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

**RC2005**

(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

**RC2487**

(2487, X'9B7') Impossible d'appeler la routine de rappel

**RC2448**

(2448, X' 990') Impossible de désenregistrer, de suspendre ou de reprendre car il n'y a pas de rappel enregistré

**RC2486**

(2486, X'9B6') Soit CallbackFunction et CallbackName ont été spécifiés sur un appel CBREG, soit CallbackFunction ou CallbackName a été spécifié mais ne correspond pas à la fonction de rappel actuellement enregistrée.

**RC2483**

(2483, X'9B3') Zone de type CallBackincorrecte.

**RC2219**

(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

**RC2444**

(2444, X'98C') Le bloc d'options est incorrect.

**RC2484**

(2484, X'9B4') Zone d'options MQCBD incorrecte.

**RC2140**

(2140, X'85C') Demande d'attente rejetée par CICS.

**RC2009**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**RC2217**

(2217, X'8A9') Non autorisé pour la connexion.

**RC2202**

(2202, X'89A') Connexion en cours de mise au repos.

- RC2203**  
(2203, X'89B') Connexion en cours d'arrêt.
- RC2207**  
(2207, X'89F') Erreur d'identificateur de corrélation.
- RC2016**  
(2016, X'7E0') Se bloque pour la file d'attente.
- RC2351**  
(2351, X'92F') Conflit entre les unités d'oeuvre globales.
- RC2186**  
(2186, X'88A') Structure des options de message Get non valide.
- RC2353**  
(2353, X'931') Descripteur en cours d'utilisation pour l'unité d'oeuvre globale.
- RC2018**  
(2018, X'7E2') Descripteur de connexion non valide.
- RC2019**  
(2019, X'7E3') Descripteur d'objet non valide.
- RC2259**  
(2259, X'8D3') Spécification de navigation incohérente.
- RC2245**  
(2245, X'8C5') Spécification d'unité d'oeuvre incohérente.
- RC2246**  
(2246, X'8C6') Message sous le curseur non valide pour la récupération.
- RC2352**  
(2352, X'930') Conflit entre l'unité d'oeuvre globale et l'unité d'oeuvre locale.
- RC2247**  
(2247, X'8C7') Options de correspondance non valides.
- RC2485**  
(2485, X'9B5') Zone de longueur MaxMsgincorrecte
- RC2026**  
(2026, X'7EA') Descripteur de message non valide.
- RC2497**  
(2497, X'9C1') Le point d'entrée de fonction spécifié est introuvable dans le module.
- RC2496**  
(2496, X'9C0') Le module a été trouvé mais son type est incorrect (32 bits ou 64 bits) ou il ne s'agit pas d'une dll valide.
- RC2495**  
(2495, X'9BF') Module introuvable dans le chemin de recherche ou non autorisé à être chargé.
- RC2206**  
(2206, X'89E') Erreur d'identificateur de message.
- RC2250**  
(2250, X'8CA') Numéro de séquence de message non valide.
- RC2331**  
(2331, X'91B') Utilisation du jeton de message non valide.
- RC2036**  
(2036, X'7F4') File d'attente non ouverte pour exploration.
- RC2037**  
(2037, X'7F5') File d'attente non ouverte pour saisie.
- RC2041**  
(2041, X'7F9') Définition d'objet modifiée depuis son ouverture.

- RC2101**  
(2101, X'835') Objet endommagé.
- RC2488**  
(2488, X'9B8') Code opération incorrect sur l'appel API
- RC2046**  
(2046, X'7FE') Options incorrectes ou incohérentes.
- RC2193**  
(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.
- RC2052**  
(2052, X'804') File d'attente supprimée.
- RC2394**  
(2394, X'95A') Type d'index incorrect pour la file d'attente.
- RC2058**  
(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.
- RC2059**  
(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.
- RC2161**  
(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.
- RC2162**  
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.
- RC2102**  
(2102, X'836') Ressources système disponibles insuffisantes.
- RC2069**  
(2069, X'815') Signal en attente pour ce descripteur.
- RC2071**  
(2071, X'817') Mémoire disponible insuffisante.
- RC2109**  
(2109, X'83D') Appel supprimé par le programme d'exit.
- RC2072**  
(2072, X'818') Prise en charge du point de synchronisation non disponible.
- RC2195**  
(2195, X'893') Une erreur inattendue s'est produite.
- RC2354**  
(2354, X'932') Echec de l'inscription dans l'unité d'oeuvre globale.
- RC2355**  
(2355, X'933') Mélange d'appels d'unité d'oeuvre non pris en charge.
- RC2255**  
(2255, X'8CF') Unité d'oeuvre non disponible pour le gestionnaire de files d'attente à utiliser.
- RC2090**  
(2090, X'82A') Intervalle d'attente non valide dans MQGMO.
- RC2256**  
(2256, X'8D0') Version MQGMO fournie non valide.
- RC2257**  
(2257, X'8D1') Version MQMD fournie non valide.
- RC2298**  
(2298, X'8FA') La fonction demandée n'est pas disponible dans l'environnement en cours.



## Déclaration RPG

```
C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQCTL(HCONN : OPERATN : PCTLOP :
                        CMPCOD : REASON)
```

La définition de prototype pour l'appel est la suivante:

```
DMQCTL          PR          EXTPROC('MQCTL')
D* Connection handle
D HCONN          10I 0 VALUE
D* Operation
D OPERATN        10I 0 VALUE
D* Control options
D PCTLOP          32A
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CompCode
D REASON          10I 0
```

IBM i

## MQDISC (gestionnaire de files d'attente de déconnexion) sous IBM i

L'appel MQDISC interrompt la connexion entre le gestionnaire de files d'attente et le programme d'application et est l'inverse de l'appel MQCONN ou MQCONNX.

- [«Syntaxe»](#), à la page 1353
- [«Remarques sur l'utilisation»](#), à la page 1353
- [«Paramètres»](#), à la page 1353
- [«Déclaration RPG»](#), à la page 1354

### Syntaxe

MQDISC (*HCONN*, *CMPCOD*, *REASON*)

### Remarques sur l'utilisation

1. Si un appel MQDISC est émis alors que l'application a toujours des objets ouverts, ces objets sont fermés par le gestionnaire de files d'attente, avec les options de fermeture définies sur CONONE.
2. Si l'application se termine avec des modifications non validées dans une unité d'oeuvre, la disposition de ces modifications dépend de la façon dont l'application se termine:
  - a. Si l'application émet l'appel MQDISC avant l'arrêt:
    - Pour une unité d'oeuvre coordonnée du gestionnaire de files d'attente, le gestionnaire de files d'attente émet l'appel MQCMIT pour le compte de l'application. L'unité de travail est validée si possible et annulée si ce n'est pas le cas.
    - Pour une unité de travail coordonnée en externe, le statut de l'unité de travail ne change pas ; toutefois, le gestionnaire de files d'attente indique que l'unité de travail doit être validée, à la demande du coordinateur de l'unité de travail.
  - b. Si l'application s'arrête normalement mais sans émettre l'appel MQDISC, l'unité de travail est annulée.
  - c. Si l'application se termine *de manière anormale* sans émettre l'appel MQDISC, l'unité de travail est annulée.

### Paramètres

L'appel MQDISC comporte les paramètres suivants:

## **HCONN (entier signé à 10 chiffres)-entrée / sortie**

Descripteur de connexion.

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *HCONN* a été renvoyée par un appel *MQCONN* ou *MQCONN*X précédent.

Une fois l'appel terminé, le gestionnaire de files d'attente définit *HCONN* sur une valeur qui n'est pas un descripteur valide pour l'environnement. Cette valeur est:

### **HCUNUH**

Descripteur de connexion inutilisable.

## **CMPCOD (entier signé à 10 chiffres)-sortie**

Code achèvement.

Il peut s'agir de :

### **CCOK**

Achèvement réussi.

### **CCWARN**

Avertissement (achèvement partiel).

### **CCFAIL**

Echec de l'appel.

## **REASON (entier signé à 10 chiffres)-sortie**

Code anomalie qualifiant *CMPCOD*.

Si *CMPCOD* est *CCOK*:

### **RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CMPCOD* est *CCFAIL*:

### **RC2219**

(2219, X'8AB') Appel *MQI* entré à nouveau avant la fin de l'appel précédent.

### **RC2009**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

### **RC2018**

(2018, X'7E2') Descripteur de connexion non valide.

### **RC2058**

(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

### **RC2059**

(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

### **RC2162**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

### **RC2102**

(2102, X'836') Ressources système disponibles insuffisantes.

### **RC2071**

(2071, X'817') Mémoire disponible insuffisante.

### **RC2195**

(2195, X'893') Une erreur inattendue s'est produite.

## **Déclaration RPG**

```
C*..1.....2.....3.....4.....5.....6.....7..
C                                CALLP      MQDISC(HCONN : CMPCOD : REASON)
```

La définition de prototype pour l'appel est la suivante:

```
D*..1.....2.....:.....3.....4.....:.....5.....:.....6.....7..
DMQDISC          PR          EXTPROC('MQDISC')
D* Connection handle
D HCONN          10I 0
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CMPCOD
D REASON          10I 0
```

## MQDLTMH (Delete message handle) sur IBM i

L'appel MQDLTMH supprime un descripteur de message et est l'inverse de l'appel MQCRTMH.

- «Syntaxe», à la page [1355](#)
- «Remarques sur l'utilisation», à la page [1355](#)
- «Paramètres», à la page [1357](#)
- «Déclaration RPG», à la page [1358](#)

### Syntaxe

((*Hconn*, *Hmsg*, *DltMsgHOpts*, *CompCode*, *Reason*) MQDLTMH

### Remarques sur l'utilisation

1. Vous pouvez utiliser cet appel uniquement lorsque le gestionnaire de files d'attente lui-même coordonne l'unité d'oeuvre. Ce nom peut être :
  - Unité de travail locale, dans laquelle les modifications affectent uniquement les ressources IBM MQ .
  - Unité de travail globale dans laquelle les modifications peuvent affecter les ressources appartenant à d'autres gestionnaires de ressources, ainsi que les ressources IBM MQ .

Pour plus de détails sur les unités d'oeuvre locales et globales, voir [«MQBEGIN \(Début de l'unité de travail\) sur IBM i»](#), à la page [1315](#).
2. Dans les environnements où le gestionnaire de files d'attente ne coordonne pas l'unité d'oeuvre, utilisez l'appel d'annulation approprié à la place de MQBACK. L'environnement peut également prendre en charge une interruption implicite provoquée par l'arrêt anormal de l'application.
  - Sous z/OS, utilisez les appels suivants:
    - Les programmes batch (y compris les programmes IMS batch DL/I) peuvent utiliser l'appel MQBACK si l'unité de travail n'affecte que les ressources IBM MQ . Toutefois, si l'unité de travail affecte à la fois les ressources IBM MQ et les ressources appartenant à d'autres gestionnaires de ressources (par exemple, Db2 ), utilisez l'appel SRRBACK fourni par le service RRS ( z/OS Recoverable Resource Service). L'appel SRRBACK annule les modifications apportées aux ressources appartenant aux gestionnaires de ressources qui ont été activés pour la coordination RRS.
    - Les applications CICS doivent utiliser la commande EXEC CICS SYNCPOINT ROLLBACK pour rétablir l'unité d'oeuvre. N'utilisez pas l'appel MQBACK pour les applications CICS .
    - Les applications IMS (autres que les programmes batch DL/I) doivent utiliser des appels IMS tels que ROLB pour rétablir l'unité d'oeuvre. N'utilisez pas l'appel MQBACK pour les applications IMS (autres que les programmes batch DL/I).
  - Sous IBM i, utilisez cet appel pour les unités de travail locales coordonnées par le gestionnaire de files d'attente. Cela signifie qu'une définition de validation ne doit pas exister au niveau du travail, c'est-à-dire que la commande STRCMTCTL avec le paramètre **CMTSCOPE(\*JOB)** ne doit pas avoir été émise pour le travail.

3. Si une application se termine avec des modifications non validées dans une unité d'oeuvre, la disposition de ces modifications varie selon que l'application se termine normalement ou anormalement. Pour plus de détails, voir les remarques sur l'utilisation dans «MQDISC (gestionnaire de files d'attente de déconnexion) sous IBM i», à la page 1353 .

4. Lorsqu'une application insère ou extrait des messages dans des groupes ou des segments de messages logiques, le gestionnaire de files d'attente conserve les informations relatives au groupe de messages et au message logique pour les derniers appels MQPUT et MQGET ayant abouti. Ces informations sont associées à l'identificateur de file d'attente et incluent les éléments suivants:

- Valeurs des zones *GroupId*, *MsgSeqNumber*, *Offset* et *MsgFlags* dans MQMD.
- Indique si le message fait partie d'une unité de travail.
- Pour l'appel MQPUT: indique si le message est persistant ou non persistant.

Le gestionnaire de files d'attente conserve trois ensembles d'informations de groupe et de segment, un ensemble pour chacun des éléments suivants:

- Dernier appel MQPUT réussi (qui peut faire partie d'une unité de travail).
- Dernier appel MQGET réussi qui a supprimé un message de la file d'attente (il peut faire partie d'une unité d'oeuvre).
- Dernier appel MQGET ayant réussi à parcourir un message dans la file d'attente (il ne peut pas faire partie d'une unité d'oeuvre).

Si l'application insère ou obtient les messages dans le cadre d'une unité de travail et que l'application annule ensuite l'unité de travail, les informations de groupe et de segment sont restaurées à la valeur qu'elle avait précédemment:

- Les informations associées à l'appel MQPUT sont restaurées à la valeur qu'elles avaient avant le premier appel MQPUT réussi pour ce descripteur de file d'attente dans l'unité de travail en cours.
- Les informations associées à l'appel MQGET sont restaurées à la valeur qu'elles avaient avant le premier appel MQGET réussi pour ce descripteur de file d'attente dans l'unité d'oeuvre en cours.

Les files d'attente qui ont été mises à jour par l'application après le démarrage de l'unité de travail, mais en dehors de la portée de l'unité de travail, ne voient pas leurs informations de groupe et de segment restaurées si l'unité de travail est annulée.

La restauration des informations de groupe et de segment à sa valeur précédente lors de l'annulation d'une unité de travail permet à l'application de répartir un grand groupe de messages ou un grand message logique composé de plusieurs segments sur plusieurs unités de travail, et de redémarrer au point approprié du groupe de messages ou du message logique si l'une des unités de travail échoue. L'utilisation de plusieurs unités d'oeuvre peut s'avérer avantageuse si le gestionnaire de files d'attente local ne dispose que d'une mémoire de file d'attente limitée. Toutefois, l'application doit conserver suffisamment d'informations pour pouvoir redémarrer l'insertion ou l'obtention de messages au point approprié en cas de défaillance du système.

Pour plus de détails sur le redémarrage au point approprié après une défaillance du système, voir l'option PMLOGO décrite dans PMOPT (entier signé à 10 chiffres) et l'option GMLOGO décrite dans GMOPT (entier signé à 10 chiffres).

Les autres remarques d'utilisation s'appliquent uniquement lorsque le gestionnaire de files d'attente coordonne les unités de travail:

5. Une unité de travail a la même portée qu'un descripteur de connexion. Tous les appels IBM MQ qui affectent une unité de travail particulière doivent être exécutés à l'aide du même descripteur de connexion. Les appels émis à l'aide d'un descripteur de connexion différent (par exemple, les appels émis par une autre application) affectent une unité d'oeuvre différente. Pour plus d'informations sur la portée des descripteurs de connexion, voir HCONN (entier signé à 10 chiffres)-output .
6. Seuls les messages insérés ou extraits dans le cadre de l'unité d'oeuvre en cours sont affectés par cet appel.
7. Une application à exécution longue qui émet des appels MQGET, MQPUT ou MQPUT1 dans une unité de travail, mais qui n'émet jamais d'appel de validation ou d'annulation, peut remplir les files d'attente

avec des messages qui ne sont pas disponibles pour d'autres applications. Pour éviter cette possibilité, l'administrateur doit définir l'attribut de gestionnaire de files d'attente **MaxUncommittedMsgs** sur une valeur suffisamment faible pour empêcher les applications en boucle indéfinie de remplir les files d'attente, mais suffisamment élevée pour permettre aux applications de messagerie attendues de fonctionner correctement.

## Paramètres

L'appel MQDLTMH comporte les paramètres suivants:

### HCONN (entier signé à 10 chiffres)-entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente.

La valeur doit correspondre au descripteur de connexion utilisé pour créer le descripteur de message spécifié dans le paramètre **HMSG**.

Si le descripteur de message a été créé à l'aide de HCUNAS, une connexion valide doit être établie sur l'unité d'exécution en supprimant le descripteur de message, sinon l'appel échoue avec RC2009.

### HMSG (entier signé à 20 chiffres)-entrée / sortie

Il s'agit du descripteur de message à supprimer. La valeur a été renvoyée par un appel MQCRTMH précédent.

Une fois l'appel terminé, le descripteur est défini sur une valeur non valide pour l'environnement. Cette valeur est:

#### HMUNUH

Descripteur de message inutilisable.

Le descripteur de message ne peut pas être supprimé si un autre appel IBM MQ en cours a été transmis avec le même descripteur de message.

### DLTOPT (MQDMHO)-entrée

Pour plus d'informations, voir [MQDMHO](#).

### CMPCOD (entier signé à 10 chiffres)-sortie

Code achèvement ; il s'agit d'un des codes suivants :

#### CCOK

Achèvement réussi.

#### CCFAIL

Echec de l'appel.

### REASON (entier signé à 10 chiffres)-sortie

Code anomalie qualifiant *CMPCOD*.

Si *CMPCOD* est CCOK:

#### RCNONE

(0, X'000') Aucun code anomalie à signaler.

Si *CMPCOD* est CCFAIL:

#### RC2204

(2204, X'089C') Carte non disponible.

#### RC2130

(2130, X'852') Impossible de charger le module de service d'adaptateur.

#### RC2157

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

#### RC2219

(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

**RC2009**

(2009, X'07D9') La connexion au gestionnaire de files d'attente a été perdue.

**RC2462**

(2462, X'099E') La structure des options de suppression de descripteur de message n'est pas valide.

**RC2460**

(2460, X'099C') Le pointeur de descripteur de message n'est pas valide.

**RC2499**

(2499, X'09C3') Descripteur de message déjà utilisé.

**RC2046**

(2046, X'07FE') Options non valides ou non cohérentes.

**RC2071**

(2071, X'817') Mémoire disponible insuffisante.

**RC2195**

(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations, voir [«Codes retour pour IBM i \(ILE RPG\)»](#), à la page 1498.

**Déclaration RPG**

```

C*.1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQDLTMH(HCONN : HMSG : DLTOPT :
                      CMPCOD : REASON)

```

La définition de prototype pour l'appel est la suivante:

```

DMQDLTMH      PR              EXTPROC('MQDLTMH')
D* Connection handle
D HCONN              10I 0 VALUE
D* Message handle
D HMSG              20I 0
D* Options that control the action of MQDLTMH
D DLTOPT            12A
D* Completion code
D CMPCOD            10I 0
D* Reason code qualifying CompCode
D REASON            10I 0

```

**MQDLTMP-Propriété de message Delete**

L'appel MQDLTMP supprime une propriété d'un descripteur de message et est l'inverse de l'appel MQSETMP.

- [«Syntaxe»](#), à la page 1358
- [«Paramètres»](#), à la page 1358
- [«Déclaration RPG»](#), à la page 1360

**Syntaxe**

(*Hconn, Hmsg, DltPropOpts, Name, CompCode, Reason*) MQDLTMP

**Paramètres**

L'appel MQDLTMP comporte les paramètres suivants:

### **HCONN (entier signé à 10 chiffres)-Entrée**

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur doit correspondre au descripteur de connexion utilisé pour créer le descripteur de message spécifié dans le paramètre **HMSG**.

Si le descripteur de message a été créé à l'aide de HCUNAS, une connexion valide doit être établie sur l'unité d'exécution en supprimant le descripteur de message, sinon l'appel échoue avec RC2009.

### **HMSG (entier signé à 20 chiffres)-entrée**

Il s'agit du descripteur de message contenant la propriété à supprimer. La valeur a été renvoyée par un appel MQCRTMH précédent.

### **DLTOPT (MQDMPO)-Entrée**

Pour plus de détails, voir le type de données MQDMPO.

### **PRNAME (MQCHARV)-entrée**

Nom de la propriété à supprimer. Pour plus d'informations sur les noms de propriété, voir Noms de propriété.

Les caractères génériques ne sont pas autorisés dans le nom de la propriété.

### **CMPCOD (entier signé à 10 chiffres)-sortie**

Code achèvement ; il s'agit d'un des codes suivants :

#### **CCOK**

Achèvement réussi.

#### **CCWARN**

Avertissement (achèvement partiel).

#### **CCFAIL**

Echec de l'appel.

### **REASON (entier signé à 10 chiffres)-sortie**

Code anomalie qualifiant *CMPCOD*.

Si *CMPCOD* est CCOK:

#### **RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CMPCOD* est CCWARN:

#### **RC2471**

(2471, X'09A7') Propriété non disponible.

#### **RC2421**

(2421, X'0975') Un dossier MQRFH2 contenant des propriétés n'a pas pu être analysé.

Si *CMPCOD* est CCFAIL:

#### **RC2204**

(2204, X'089C') Carte non disponible.

#### **RC2130**

(2130, X'0852') Impossible de charger le module de service de l'adaptateur.

#### **RC2157**

(2157, X'086D') Les ASID principal et principal diffèrent.

#### **RC2219**

(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

#### **RC2009**

(2009, X'07D9') La connexion au gestionnaire de files d'attente a été perdue.

**RC2481**

(2481, X'09B1') La structure des options de suppression des propriétés de message n'est pas valide.

**RC2460**

(2460, X'099C') Descripteur de message non valide.

**RC2499**

(2499, X'09C3') Descripteur de message déjà utilisé.

**RC2046**

(2046, X'07FE') Options non valides ou non cohérentes.

**RC2442**

(2442, X'098A') Nom de propriété non valide.

**RC2111**

(2111, X'083F') Identificateur de jeu de caractères codés de nom de propriété non valide.

**RC2195**

(2195, X'0893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir [Codes achèvement et raison d'API](#).

**Déclaration RPG**

```
C*.1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQDLTMP(HCONN : HMSG : DLTOPT :
                      PRNAME : CMPCOD : REASON)
```

La définition de prototype pour l'appel est la suivante:

```
DMQDLTMP          PR          EXTPROC('MQDLTMP')
D* Connection handle
D HCONN           10I 0 VALUE
D* Message handle
D HMSG            20I 0 VALUE
D* Options that control the action of MQDLTMP
D DLTOPT          12A
D* Property name
D PRNAME          32A
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CompCode
D REASON          10I 0
```

**IBM i MQGET (extraction de message) sur IBM i**

L'appel MQGET extrait un message d'une file d'attente locale qui a été ouverte à l'aide de l'appel MQOPEN.

- [«Syntaxe», à la page 1360](#)
- [«Remarques sur l'utilisation», à la page 1361](#)
- [«Paramètres», à la page 1364](#)
- [«Déclaration RPG», à la page 1368](#)

**Syntaxe**

MQGET (HCONN, HOBJ, MSGDSC, GMO, BUFLen, BUFFER, DATLEN, CMPCOD, REASON)



## Remarques sur l'utilisation

1. Le message extrait est normalement supprimé de la file d'attente. Cette suppression peut se produire dans le cadre de l'appel MQGET lui-même ou dans le cadre d'un point de synchronisation. La suppression de message n'est pas effectuée si une option GMBRWF ou GMBRWN est indiquée au paramètre **GMO** (voir la zone *GMOPT* décrite dans «MQGMO (options d'extraction de message) sous IBM i», à la page 1127).
2. Si l'option GMLK est spécifiée avec l'une des options de navigation, le message consulté est verrouillé de sorte qu'il ne soit visible que par ce descripteur.

Si l'option GMUNLK est spécifiée, un message précédemment verrouillé est déverrouillé. Aucun message n'est extrait dans ce cas, et les paramètres **MSGDSC**, **BUFLEN**, **BUFFER** et **DATLEN** ne sont pas vérifiés ou modifiés.

3. Si l'application émettrice de l'appel MQGET s'exécute en tant que IBM MQ MQI client, il est possible que le message extrait soit perdu si, lors du traitement de l'appel MQGET, IBM MQ MQI client s'arrête de manière anormale ou que la connexion client est interrompue. Cela est dû au fait que le substitut qui s'exécute sur la plateforme du gestionnaire de files d'attente et qui émet l'appel MQGET pour le compte du client ne peut pas détecter la perte du client tant que le substitut n'est pas sur le point de renvoyer le message au client ; c'est une fois que le message a été supprimé de la file d'attente. Cela peut se produire pour les messages persistants et les messages non persistants.

Le risque de perte de messages de cette manière peut être éliminé en extrayant toujours les messages dans les unités de travail (c'est-à-dire en spécifiant l'option GMSYP sur l'appel MQGET et en utilisant les appels MQCMIT ou MQBACK pour valider ou supprimer l'unité de travail lorsque le traitement du message est terminé). Si GMSYP est spécifié et que le client s'arrête de manière anormale ou que la connexion est interrompue, le substitut annule l'unité d'oeuvre sur le gestionnaire de files d'attente et le message est rétabli dans la file d'attente.

En principe, la même situation peut se produire avec les applications qui s'exécutent sur la plateforme du gestionnaire de files d'attente, mais dans ce cas, la fenêtre pendant laquelle un message peut être perdu est petite. Toutefois, comme avec IBM MQ MQI clients, le risque peut être éliminé en récupérant le message dans une unité de travail.

4. Si une application place une séquence de messages dans une file d'attente particulière au sein d'une unité d'oeuvre unique et qu'elle valide ensuite cette unité d'oeuvre correctement, les messages deviennent disponibles pour extraction, comme suit :
  - Si la file d'attente est une *file d'attente non partagée* (c'est-à-dire une file d'attente locale), tous les messages de l'unité d'oeuvre deviennent disponibles en même temps.
  - Si la file d'attente est une *file d'attente partagée*, les messages de l'unité d'oeuvre sont disponibles dans l'ordre dans lequel ils ont été insérés, mais pas tous en même temps. Lorsque le système est fortement chargé, il est possible que le premier message de l'unité de travail soit extrait avec succès, mais que l'appel MQGET pour le deuxième message ou le message suivant de l'unité de travail échoue avec RC2033. Si cela se produit, l'application doit attendre un court moment, puis relancer l'opération.
5. Si une application place une séquence de messages dans la même file d'attente sans utiliser de groupes de messages, l'ordre de ces messages est préservé sous réserve que certaines conditions soient satisfaites. Pour plus de détails, voir les remarques sur l'utilisation dans la description de l'appel MQPUT. Si les conditions sont satisfaites, les messages sont présentés à l'application réceptrice dans l'ordre dans lequel ils ont été envoyés, sous les réserves suivantes :
  - Un seul récepteur extrait des messages de la file d'attente.

Si deux applications ou plus extraient des messages de la file d'attente, elles doivent convenir avec l'émetteur du mécanisme à utiliser pour identifier les messages faisant partie d'une séquence. Par exemple, l'expéditeur peut définir toutes les zones MDCID dans les messages d'une séquence sur une valeur unique à cette séquence de messages.
  - Le récepteur ne modifie pas délibérément l'ordre d'extraction, par exemple en spécifiant un MDMID ou un MDCID particulier.

Si l'application émettrice place les messages en tant que groupe de messages, les messages sont présentés à l'application réceptrice dans l'ordre correct si l'application réceptrice spécifie l'option GMLOGO sur l'appel MQGET. Pour plus d'informations sur les groupes de messages, voir :

- Zone MDMFL dans MQMD
  - Option PMLOGO dans MQPMO
  - Option GMLOGO dans MQGMO
6. Les applications testent le code retour FBQUIT dans la zone MDFB du paramètre **MSGDSC** . Si cette valeur est trouvée, l'application se termine. Pour plus d'informations, voir la zone MDFB décrite dans «MQMD (descripteur de message) sous IBM i», à la page 1162 .
  7. Si la file d'attente identifiée par HOBj a été ouverte avec l'option OOSAVA et que le code achèvement de l'appel MQGET est CCOK ou CCWARN, le contexte associé à l'identificateur de file d'attente HOBj est défini sur le contexte du message qui a été extrait (sauf si l'option GMBRWF ou GMBRWN est définie, auquel cas le contexte est marqué comme non disponible). Ce contexte peut être utilisé lors d'un appel MQPUT ou MQPUT1 ultérieur en spécifiant les options PMPASI ou PMPASA. Cela permet de transférer en intégralité ou en partie le contexte du message reçu vers un autre message (par exemple, lorsque le message est réacheminé vers une autre file d'attente). Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).
  8. Si l'option GMCONV est incluse dans le paramètre **GMO** , les données du message d'application sont converties en la représentation demandée par l'application de réception, avant que les données ne soient placées dans le paramètre **BUFFER** :
    - La zone MDFMT dans les informations de contrôle du message identifie la structure des données d'application et les zones MDCSI et MDENC dans les informations de contrôle du message spécifient son identificateur de jeu de caractères et son codage.
    - L'application émettant l'appel MQGET spécifie dans les zones MDCSI et MDENC du paramètre **MSGDSC** l'identificateur de jeu de caractères et le codage dans lesquels les données de message d'application doivent être converties.

Lorsque la conversion des données de message est nécessaire, elle est effectuée soit par le gestionnaire de files d'attente lui-même, soit par un exit écrit par l'utilisateur, en fonction de la valeur de la zone MDFMT dans les informations de contrôle du message:

- Les formats suivants sont convertis automatiquement par le gestionnaire de files d'attente ; ces formats sont appelés formats "intégrés":

FMADMN	FMMDE
modulation de fréquenceCICS	FMPCF
FMCMD1	FMRMH
FMCMD2	FMRFH
FMDLH	FMRFH2
FMDH	FSTR
FMEVNT	FMTM
modulation de fréquenceIMS	FMXQH
FMIMVS	

- Le nom de format FMNONE est une valeur spéciale qui indique que la nature des données du message n'est pas définie. En conséquence, le gestionnaire de files d'attente ne tente pas la conversion lorsque le message est extrait de la file d'attente.

**Remarque :** Si GMCONV est spécifié dans l'appel MQGET pour un message dont le nom de format est FMNONE et que le jeu de caractères ou le codage du message est différent de celui spécifié dans le paramètre **MSGDSC** , le message est toujours renvoyé dans le paramètre **BUFFER** (en supposant qu'aucune autre erreur ne se soit produite), mais l'appel se termine avec le code achèvement CCWARN et le code anomalie RC2110.

FMNONE peut être utilisé soit lorsque la nature des données de message signifie qu'elle ne nécessite pas de conversion, soit lorsque les applications d'envoi et de réception ont convenu entre elles de la forme sous laquelle les données de message doivent être envoyées.

- Tous les autres noms de format entraînent la transmission du message à un exit écrit par l'utilisateur pour la conversion. L'exit porte le même nom que le format, hormis des ajouts spécifiques à l'environnement. Les noms de format spécifiés par l'utilisateur ne doivent pas commencer par les lettres "MQ", car ces noms peuvent entrer en conflit avec les noms de format pris en charge ultérieurement.

Les données utilisateur du message peuvent être converties d'un jeu de caractères pris en charge à l'autre et d'un codage à l'autre. Toutefois, sachez que si le message contient une ou plusieurs structures d'en-tête IBM MQ, le message ne peut pas être converti depuis ou vers un jeu de caractères comportant des caractères codés sur deux octets ou sur plusieurs octets pour l'un des caractères admis dans les noms de file d'attente. Le code anomalie RC2111 ou RC2115 est généré si une tentative est effectuée et le message est renvoyé non converti. Le jeu de caractères Unicode UTF-16 est un exemple de ce type de jeu de caractères.

Renvoyé par MQGET, le code anomalie suivant indique que la conversion du message a abouti :

- RCNONE

Le code anomalie suivant indique que le message a peut-être été converti ; l'application doit vérifier les zones MDCSI et MDENC du paramètre **MSGDSC** pour le savoir:

- RC2079

Tous les autres codes anomalie indiquent que le message n'a pas été converti.

**Remarque :** L'interprétation du code anomalie décrit dans cet exemple est vraie pour les conversions effectuées par des exits écrits par l'utilisateur uniquement si l'exit est conforme aux instructions de traitement.

9. Pour les formats intégrés répertoriés précédemment, le gestionnaire de files d'attente peut effectuer la conversion par défaut des chaînes de caractères du message lorsque l'option GMCONV est spécifiée. La conversion par défaut permet au gestionnaire de files d'attente d'utiliser, lors de la conversion des données de type chaîne, un jeu de caractères par défaut spécifié par l'installation qui se rapproche du jeu de caractères réel. Par conséquent, l'appel MQGET peut aboutir avec le code achèvement CCOK, au lieu de se terminer avec CCWARN et le code anomalie RC2111 ou RC2115.

**Remarque :** L'utilisation d'un jeu de caractères approximatif pour la conversion des données de type chaîne donne lieu à la conversion éventuellement incorrecte de certains caractères. Cela peut être évité en utilisant dans la chaîne uniquement des caractères qui sont communs au jeu de caractères réel et au jeu de caractères par défaut.

La conversion par défaut s'applique aux données de message d'application et aux zones alphanumériques des structures MQMD et MQMDE :

- La conversion par défaut des données de message d'application se produit uniquement lorsque toutes les instructions suivantes sont vraies:
  - L'application spécifie GMCONV.
  - Le message contient des données qui doivent être converties depuis ou vers un jeu de caractères qui n'est pas pris en charge.
  - La conversion par défaut a été activée lors de l'installation ou du redémarrage du gestionnaire de files d'attente.
- La conversion par défaut des zones alphanumériques dans les structures MQMD et MQMDE a lieu selon les besoins, si la conversion par défaut est activée pour le gestionnaire de files d'attente. La conversion est effectuée même si l'option GMCONV n'est pas spécifiée par l'application dans l'appel MQGET.

10. Le paramètre **BUFFER** indiqué dans l'exemple de programmation RPG est déclaré sous la forme d'une chaîne, ce qui limite la longueur maximale du paramètre à 256 octets. Si une mémoire tampon plus

importante est requise, le paramètre doit être déclaré en tant que structure ou zone dans un fichier physique.

La déclaration du paramètre en tant que structure augmente la longueur maximale possible à 9999 octets, tandis que la déclaration du paramètre en tant que zone dans un fichier physique augmente la longueur maximale possible à environ 32 Ko.

## Paramètres

L'appel MQGET comporte les paramètres suivants:

### **HCONN (entier signé à 10 chiffres)-entrée**

Descripteur de connexion.

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de HCONN a été renvoyée par un appel MQCONN ou MQCONNX précédent.

### **HOBJ (entier signé à 10 chiffres)-entrée**

Descripteur d'objet.

Ce descripteur représente la file d'attente à partir de laquelle un message est récupéré. La valeur de HOBJ a été renvoyée par un appel MQOPEN précédent. La file d'attente doit avoir été ouverte avec une ou plusieurs des options suivantes (voir [«MQOPEN \(objet ouvert\) sur IBM i»](#), à la page 1386 pour plus de détails):

- OOINPS
- OOINPX
- OOINPQ
- OOBW

### **MSGDSC (MQMD)-entrée / sortie**

Descripteur de message.

Cette structure décrit les attributs du message requis ainsi que ceux du message récupéré. Pour plus d'informations, voir [«MQMD \(descripteur de message\) sous IBM i»](#), à la page 1162.

Si BUFLen est inférieur à la longueur du message, MSGDSC est toujours entré par le gestionnaire de files d'attente, que GMATM soit spécifié ou non dans le paramètre **GMO** (voir la zone GMOPT décrite dans [«MQGMO \(options d'extraction de message\) sous IBM i»](#), à la page 1127).

Si l'application fournit un MQMD version-1, le message renvoyé a un MQMDE préfixé aux données du message d'application, mais uniquement si une ou plusieurs des zones du MQMDE ont une valeur autre que la valeur par défaut. Si toutes les zones du MQMDE ont des valeurs par défaut, le MQMDE est omis. Un nom de format FMMDE dans la zone MDFMT de MQMD indique qu'un MQMDE est présent.

### **OGM (MQGMO)-entrée/sortie**

Options qui contrôlent l'action de MQGET.

Pour plus d'informations, voir [«MQGMO \(options d'extraction de message\) sous IBM i»](#), à la page 1127.

### **BUFLen (entier signé à 10 chiffres)-entrée**

Longueur en octets de la zone BUFFER .

La valeur zéro peut être indiquée pour les messages qui n'ont pas de données ou si le message doit être supprimé de la file d'attente et que les données doivent être supprimées (GMATM doit être indiqué dans ce cas).

**Remarque :** La longueur du message le plus long qu'il est possible de lire à partir de la file d'attente est donnée par l'attribut de file d'attente **MaxMsgLength** ; voir [«Attributs des files d'attente»](#), à la page 1435.

## **BUFFER (chaîne de 1 octet x BUFLen)-sortie**

Zone devant contenir les données du message.

La mémoire tampon doit être alignée sur une limite appropriée à la nature des données du message. L'alignement à 4 octets doit être adapté à la plupart des messages (y compris les messages contenant des structures d'en-tête IBM MQ), mais certains messages peuvent nécessiter un alignement plus strict. Par exemple, un message contenant un entier binaire de 64 bits peut nécessiter un alignement de 8 octets.

Si BUFLen est inférieur à la longueur du message, la plus grande partie possible du message est déplacée dans BUFFER ; cela se produit si GMATM est spécifié sur le paramètre **GMO** (voir la zone **GMOPT** décrite dans «MQGMO (options d'extraction de message) sous IBM i», à la page 1127 pour plus d'informations).

Le jeu de caractères et le codage des données dans **BUFFER** sont fournis par les zones MDCSI et MDENC renvoyées dans le paramètre **MSGDSC**. Si ces valeurs sont différentes des valeurs requises par le récepteur, ce dernier doit convertir les données de message d'application dans le jeu de caractères et le codage requis. L'option GMCONV peut être utilisée avec un exit écrit par l'utilisateur pour effectuer la conversion des données de message (voir «MQGMO (options d'extraction de message) sous IBM i», à la page 1127 pour plus de détails sur cette option).

**Remarque :** Tous les autres paramètres de l'appel MQGET se trouvent dans le jeu de caractères et le codage du gestionnaire de files d'attente local (indiqués par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et ENNAT).

Si l'appel échoue, le contenu de la mémoire tampon peut avoir changé.

## **DATLEN (entier signé à 10 chiffres)-sortie**

Longueur du message.

Il s'agit de la longueur en octets des données d'application du message. Si la longueur de ce message est supérieure à BUFLen, seuls BUFLen octets sont renvoyés dans le paramètre **BUFFER** (c'est-à-dire que le message est tronqué). Si la valeur est zéro, cela signifie que le message ne contient pas de données d'application.

Si BUFLen est inférieur à la longueur du message, DATLEN est toujours entré par le gestionnaire de files d'attente, que GMATM soit spécifié ou non dans le paramètre **GMO** (voir la zone **GMOPT** décrite dans «MQGMO (options d'extraction de message) sous IBM i», à la page 1127 pour plus d'informations). L'application peut ainsi choisir la taille de la mémoire tampon requise pour contenir les données de message, puis exécuter de nouveau l'appel avec une taille de mémoire tampon appropriée.

Toutefois, si l'option GMCONV est spécifiée et que les données de message converties sont trop longues pour tenir dans BUFFER, la valeur renvoyée pour DATLEN est:

- Longueur des données non converties, pour les formats définis par le gestionnaire de files d'attente.

Dans ce cas, si la nature des données provoque son extension lors de la conversion, l'application doit allouer une mémoire tampon supérieure à la valeur renvoyée par le gestionnaire de files d'attente pour DATLEN.

- la valeur renvoyée par l'exit de conversion de données pour les formats d'application.

## **CMPCOD (entier signé à 10 chiffres)-sortie**

Code achèvement.

Il peut s'agir de :

### **CCOK**

Achèvement réussi.

### **CCWARN**

Avertissement (achèvement partiel).

**CCFAIL**

Echec de l'appel.

**REASON (entier signé à 10 chiffres)-sortie**

Code anomalie qualifiant CMPCOD.

Les codes anomalie suivants sont ceux que le gestionnaire de files d'attente peut renvoyer pour le paramètre **REASON** . Si l'application spécifie l'option GMCONV et qu'un exit écrit par l'utilisateur est appelé pour convertir une partie ou la totalité des données de message, c'est l'exit qui détermine la valeur renvoyée pour le paramètre **REASON** . Par conséquent, des valeurs autres que les valeurs documentées plus loin dans cette section sont possibles.

Si CMPCOD est CCOK:

**RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si CMPCOD est CCWARN:

**RC2120**

(2120, X'848') Données converties trop volumineuses pour la mémoire tampon.

**RC2190**

(2190, X'88E') Chaîne convertie trop volumineuse pour la zone.

**RC2150**

(2150, X'866') Chaîne DBCS non valide.

**RC2110**

(2110, X'83E') Format de message non valide.

**RC2243**

(2243, X'8C3') Les segments de message possèdent des CCSID différents.

**RC2244**

(2244, X'8C4') Les segments de message possèdent des codages différents.

**RC2209**

(2209, X'8A1') Aucun message verrouillé.

**RC2119**

(2119, X'847') Données de message non converties.

**RC2272**

(2272, X'8E0') Données de message partiellement converties.

**RC2145**

(2145, X'861') Paramètre de tampon source non valide.

**RC2111**

(2111, X'83F') Identificateur de jeu de caractères codés source non valide.

**RC2113**

(2113, X'841') Codage décimal condensé dans le message non reconnu.

**RC2114**

(2114, X'842') Codage à virgule flottante dans le message non reconnu.

**RC2112**

(2112, X'840') Codage d'entier source non reconnu.

**RC2143**

(2143, X'85F') Paramètre de longueur source non valide.

**RC2146**

(2146, X'862') Paramètre de tampon cible non valide.

**RC2115**

(2115, X'843') Identificateur de jeu de caractères codés cible non valide.

**RC2117**

(2117, X'845') Codage décimal condensé spécifié par le récepteur non reconnu.

**RC2118**

(2118, X'846') Codage à virgule flottante spécifié par le récepteur non reconnu.

**RC2116**

(2116, X'844') Codage d'entier cible non reconnu.

**RC2079**

(2079, X'81F') Message tronqué renvoyé (traitement terminé).

**RC2080**

(2080, X'820') Message tronqué renvoyé (traitement non terminé).

Si CMPCOD est CCFAIL:

**RC2004**

(2004, X'7D4') Paramètre de tampon non valide.

**RC2005**

(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

**RC2219**

(2219, X'8AB') Appel MQI entré à nouveau avant la fin de l'appel précédent.

**RC2009**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**RC2010**

(2010, X'7DA') Paramètre de longueur des données non valide.

**RC2016**

(2016, X'7E0') Se bloque pour la file d'attente.

**RC2186**

(2186, X'88A') Structure des options de message Get non valide.

**RC2018**

(2018, X'7E2') Descripteur de connexion non valide.

**RC2019**

(2019, X'7E3') Descripteur d'objet non valide.

**RC2241**

(2241, X'8C1') Groupe de messages non complet.

**RC2242**

(2242, X'8C2') Message logique non complet.

**RC2259**

(2259, X'8D3') Spécification de navigation incohérente.

**RC2245**

(2245, X'8C5') Spécification d'unité d'oeuvre incohérente.

**RC2246**

(2246, X'8C6') Message sous le curseur non valide pour la récupération.

**RC2247**

(2247, X'8C7') Options de correspondance non valides.

**RC2026**

(2026, X'7EA') Descripteur de message non valide.

**RC2250**

(2250, X'8CA') Numéro de séquence de message non valide.

**RC2033**

(2033, X'7F1') Aucun message disponible.

**RC2034**

(2034, X'7F2') Curseur d'exploration non positionné sur le message.

**RC2036**

(2036, X'7F4') File d'attente non ouverte pour exploration.

**RC2037**

(2037, X'7F5') File d'attente non ouverte pour saisie.

**RC2041**

(2041, X'7F9') Définition d'objet modifiée depuis son ouverture.

**RC2101**

(2101, X'835') Objet endommagé.

**RC2046**

(2046, X'7FE') Options incorrectes ou incohérentes.

**RC2052**

(2052, X'804') File d'attente supprimée.

**RC2058**

(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

**RC2059**

(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

**RC2161**

(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.

**RC2162**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**RC2102**

(2102, X'836') Ressources système disponibles insuffisantes.

**RC2071**

(2071, X'817') Mémoire disponible insuffisante.

**RC2024**

(2024, X'7E8') Aucun autre message pouvant être traité au sein de l'unité d'oeuvre en cours.

**RC2072**

(2072, X'818') Prise en charge du point de synchronisation non disponible.

**RC2195**

(2195, X'893') Une erreur inattendue s'est produite.

**RC2255**

(2255, X'8CF') Unité d'oeuvre non disponible pour le gestionnaire de files d'attente à utiliser.

**RC2090**

(2090, X'82A') Intervalle d'attente non valide dans MQGMO.

**RC2256**

(2256, X'8D0') Version MQGMO fournie non valide.

**RC2257**

(2257, X'8D1') Version MQMD fournie non valide.

**Déclaration RPG**

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQGET(HCONN : HOBJ : MSGDSC : GMO :
C                               BUFLN : BUFFER : DATLEN :
C                               CMPCOD : REASON)

```

La définition de prototype pour l'appel est la suivante:

```

D*..1.....2.....3.....4.....5.....6.....7..
DMQGET          PR          EXTPROC('MQGET')
D* Connection handle
D HCONN          10I 0 VALUE
D* Object handle
D HOBJ          10I 0 VALUE
D* Message descriptor
D MSGDSC          364A

```



```

D* Options that control the action of MQGET
D GMO 112A
D* Length in bytes of the Buffer area
D BUFLN 10I 0 VALUE
D* Area to contain the message data
D BUFFER * VALUE
D* Length of the message
D DATLEN 10I 0
D* Completion code
D CMPCOD 10I 0
D* Reason code qualifying CMPCOD
D REASON 10I 0

```

## IBM i MQINQ (interrogation des attributs d'objet) sur IBM i

L'appel MQINQ renvoie un tableau d'entiers et un ensemble de chaînes de caractères contenant les attributs d'un objet.

Les types d'objet suivants sont valides:

- File d'attente
- Liste de noms
- Définition de processus
- Gestionnaire de files d'attente
- [«Syntaxe», à la page 1369](#)
- [«Remarques sur l'utilisation», à la page 1369](#)
- [«Paramètres», à la page 1370](#)
- [«Déclaration RPG», à la page 1377](#)

### Syntaxe

MQINQ (*HCONN, HOBJ, SELCNT, SELS, IACNT, INTATR, CALEN, CHRATR, CMPCOD, REASON*)

### Remarques sur l'utilisation

1. Les valeurs renvoyées sont un instantané des attributs sélectionnés. Il n'est pas garanti que les attributs ne soient pas modifiés avant que l'application puisse agir sur les valeurs renvoyées.
2. Lorsque vous ouvrez une file d'attente modèle, une file d'attente locale dynamique est créée. Cela est vrai même si vous ouvrez la file d'attente modèle pour en savoir plus sur ses attributs.

Les attributs de la file d'attente dynamique (avec certaines exceptions) sont identiques à ceux de la file d'attente modèle au moment de la création de la file d'attente dynamique. Si vous utilisez ensuite l'appel MQINQ sur cette file d'attente, le gestionnaire de files d'attente renvoie les attributs de la file d'attente dynamique et non ceux de la file d'attente modèle. Voir le [Tableau 1](#) pour plus de détails sur les attributs de la file d'attente modèle qui sont hérités par la file d'attente dynamique.

3. Si l'objet demandé est une file d'attente alias, les valeurs d'attribut renvoyées par l'appel MQINQ sont celles de la file d'attente alias et non celles de la file d'attente de base dans laquelle l'alias est résolu.
4. Si l'objet demandé est une file d'attente de cluster, les attributs pouvant être demandé dépendent de la manière dont la file d'attente est ouverte:
  - Si la file d'attente de cluster est ouverte pour l'interrogation plus une ou plusieurs entrées, parcourez ou définissez, il doit y avoir une instance locale de la file d'attente de cluster pour que l'ouverture aboutisse. Dans ce cas, les attributs qui peuvent être renseignés sont ceux qui sont valides pour les files d'attente locales.
  - Si la file d'attente de cluster est ouverte pour l'interrogation seule ou pour l'interrogation et la sortie, seuls les attributs suivants peuvent être utilisés ; l'attribut **QType** a la valeur QTCLUS dans ce cas:
    - CAQD
    - CAQN

- IADBND
- IADPER
- IADPRI
- IAIPUT
- IAQTYP

Si la file d'attente de cluster est ouverte sans liaison fixe (c'est-à-dire, OOBNDN spécifié dans l'appel MQOPEN ou OOBNDQ spécifié lorsque l'attribut **DefBind** a la valeur BNDNOT), les appels MQINQ successifs de la file d'attente peuvent interroger différentes instances de la file d'attente de cluster, bien que toutes les instances aient généralement les mêmes valeurs d'attribut.

Pour plus d'informations sur les files d'attente de cluster, voir [Configuration d'un cluster de gestionnaires de files d'attente](#).

5. Si un certain nombre d'attributs doivent être demandés et que certains d'entre eux doivent être définis à l'aide de l'appel MQSET, il peut être utile de positionner au début des tableaux de sélecteurs les attributs à définir, de sorte que les mêmes tableaux (avec des nombres réduits) puissent être utilisés pour MQSET.
6. Si plusieurs situations d'avertissement se produisent (voir le paramètre **CMPCOD**), le code anomalie renvoyé est le *premier* de la liste suivante qui s'applique:
  - a. RC2068
  - b. RC2022
  - c. RC2008
7. Pour plus d'informations sur les attributs d'objet, voir:
  - «Attributs des files d'attente», à la page 1435
  - «Attributs des listes de noms», à la page 1466
  - «Attributs des définitions de processus sous IBM i», à la page 1467
  - «Attributs du gestionnaire de files d'attente sous IBM i», à la page 1469
8. Une nouvelle file d'attente locale SYSTEM.ADMIN.COMMAND.EVENT est utilisée pour la mise en file d'attente des messages générés chaque fois que des commandes sont émises. Les messages sont placés dans cette file d'attente pour la plupart des commandes, en fonction de la manière dont l'attribut de gestionnaire de files d'attente CMDEV est défini:
  - ENABLED-Les messages d'événement de commande sont générés et placés dans la file d'attente pour toutes les commandes ayant abouti.
  - Les messages d'événement NODISPLAY-command sont générés et placés dans la file d'attente pour toutes les commandes ayant abouti autres que la commande DISPLAY (MQSC) et la commande Inquire (PCF).
  - DISABLED-Les messages d'événement de commande ne sont pas générés (il s'agit de la valeur par défaut initiale du gestionnaire de files d'attente).

## Paramètres

L'appel MQINQ comporte les paramètres suivants:

### **HCONN (entier signé à 10 chiffres)-entrée**

Descripteur de connexion.

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *HCONN* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

### **HOBJ (entier signé à 10 chiffres)-entrée**

Descripteur d'objet.

Ce descripteur représente l'objet (de tout type) avec les attributs requis. Le descripteur doit avoir été renvoyé par un appel MOPEN précédent qui a indiqué l'option OINQ.

### **SELCNT (entier signé à 10 chiffres)-entrée**

Nombre de sélecteurs.

Nombre de sélecteurs fournis dans le tableau SELS . Il s'agit du nombre d'attributs à renvoyer. Zéro est une valeur valide. Le nombre maximal autorisé est 256.

### **SELS (entier signé à 10 chiffres x SELCNT)-entrée**

Tableau de sélecteurs d'attribut.

Il s'agit d'un tableau de sélecteurs d'attribut **SELCNT** ; chaque sélecteur identifie un attribut (entier ou caractère) avec une valeur obligatoire.

Chaque sélecteur doit être valide pour le type d'objet que *HOB*J représente, sinon l'appel échoue avec le code achèvement CCFAIL et le code anomalie RC2067.

Dans le cas particulier des files d'attente:

- Si le sélecteur n'est pas valide pour les files d'attente de type *any* , l'appel échoue avec le code achèvement CCFAIL et le code anomalie RC2067.
- Si le sélecteur est applicable uniquement aux files d'attente de type ou de types autres que celui de l'objet, l'appel aboutit avec le code achèvement CCWARN et le code anomalie RC2068.
- Si la file d'attente en cours d'interrogation est une file d'attente de cluster, les sélecteurs valides dépendent de la manière dont la file d'attente a été résolue ; voir la remarque d'utilisation 4 pour plus de détails.

Les sélecteurs peuvent être spécifiés dans n'importe quel ordre. Les valeurs d'attribut qui correspondent à des sélecteurs d'attribut d'entier (IA\* selectors) sont renvoyées dans *INTATR* dans l'ordre dans lequel ces sélecteurs apparaissent dans *SELS*. Les valeurs d'attribut qui correspondent aux sélecteurs d'attribut de caractères (sélecteurs CA\*) sont renvoyées dans *CHRATR* dans l'ordre dans lequel ces sélecteurs apparaissent. Les sélecteurs A\* peuvent être entrelacés avec les sélecteurs CA\* ; seul l'ordre relatif au sein de chaque type est important.

#### **Remarque :**

1. Les sélecteurs d'attribut d'entier et de caractère sont alloués dans deux plages différentes ; les sélecteurs IA\* résident dans la plage IAFRST à IALAST, et les sélecteurs CA\* dans la plage CAFRST à CALAST.

Pour chaque plage, les constantes IALSTU et CALSTU définissent la valeur la plus élevée acceptée par le gestionnaire de files d'attente.

2. Si tous les sélecteurs IA\* apparaissent en premier, les mêmes numéros d'éléments peuvent être utilisés pour adresser les éléments correspondants dans les tableaux *SELS* et *INTATR* .

Les attributs que vous pouvez consulter sont répertoriés dans les tableaux suivants. Pour les sélecteurs CA\*, la constante qui définit la longueur en octets de la chaîne résultante dans *CHRATR* est indiquée entre parenthèses.

<i>Tableau 746. Sélecteurs d'attributs MQINQ pour les files d'attente</i>		
<b>Sélecteur</b>	<b>Description</b>	<b>Remarque</b>
CAALTD	Date de la dernière modification (LNDATE).	1
CAALTT	Heure de la dernière modification (LNTIME).	1
CABRQN	Nom de remise en file d'attente (LNQN) trop élevé.	5
CABASQ	Nom de la file d'attente dont l'alias est résolu en (LNQN).	
CACFSN	Nom de structure d'unité de couplage (LNCFSN).	3

<i>Tableau 746. Sélecteurs d'attributs MQINQ pour les files d'attente (suite)</i>		
<b>Sélecteur</b>	<b>Description</b>	<b>Remarque</b>
CACLN	Nom de grappe (LNCLUN).	1
CACLNL	Liste de noms de cluster (LNNLN).	1
CACRDT	Date de création de la file d'attente (LNCRTD).	
CACRTT	Heure de création de la file d'attente (LNCRTT).	
CAINIQ	Nom de la file d'attente d'initialisation (LNQN).	
CAPRON	Nom de la définition de processus (LNPRON).	
CAQD	Description de file d'attente (LNQD).	
CAQN	Nom de file d'attente (LNQN).	
Réseau CARQMN	Nom du gestionnaire de files d'attente éloignées (LNQMN).	
CARQN	Nom de la file d'attente éloignée connue sur le gestionnaire de files d'attente éloignées (LNQN).	
CATRGD	Données de déclenchement (LNTRGD).	5
CAXQN	Nom de la file d'attente de transmission (LNQN).	
IABTHR	Seuil d'annulation.	5
IACDEP	Nombre de messages dans la file d'attente.	
IADBND	Liaison par défaut.	1
IADINP	Option open-for-input par défaut.	5
IADPER	Persistance de message par défaut.	
IADPRI	Priorité de message par défaut.	5
IADEFT	Type de définition de file d'attente.	
IADIST	Prise en charge de la liste de distribution.	2
IAHGB	Indique si le nombre d'annulations doit être durci.	5
IAIGET	Indique si les opérations d'extraction sont autorisées.	
IAIPUT	Indique si les opérations d'insertion sont autorisées.	
IAMLEN	Longueur maximale des messages.	
IAMDEP	Nombre maximal de messages admis dans la file d'attente.	
IAMDS	Indique si la priorité du message est pertinente.	5
IAOCI	Nombre d'appels MQOPEN dont la file d'attente est ouverte pour l'entrée.	
IAOOC	Nombre d'appels MQOPEN dont la file d'attente est ouverte pour la sortie.	
IAQDHE	Attribut de contrôle pour les événements de longueur élevée de file d'attente.	4, 5
IAQDHL	Limite supérieure de la longueur de la file d'attente.	4, 5
IAQDLE	Attribut de contrôle pour les événements de longueur faible de la file d'attente.	4, 5
DLL AQIQ	Limite inférieure de la longueur de la file d'attente.	4, 5
IAQDME	Attribut de contrôle des événements de longueur maximale de la file d'attente.	4, 5

Sélecteur	Description	Remarque
IAQSI	Limite de l'intervalle de service de file d'attente.	4, 5
IAQSIE	Attribut de contrôle des événements d'intervalle de service de file d'attente.	4, 5
IAQTYP	Type de file d'attente.	
ID SGD	Agencement de groupe de partage de files d'attente.	3
IARINT	Intervalle de conservation de la file d'attente.	5
IASCOP	Portée de la définition de file d'attente.	4, 5
IASHAR	Indique si la file d'attente peut être partagée pour l'entrée.	
IATRGC	Contrôle du déclencheur.	
IATRGD	Longueur de déclenchement.	5
IATRGP	Priorité de message de seuil pour les déclencheurs.	5
IATRGT	Type de déclencheur.	
IAUSAG	Utilisation.	
CLWLUSEQ	Utilisez des files d'attente éloignées.	

**Remarque :**

1. Pris en charge sur les plateformes suivantes:

-  AIX
-  IBM i
-  Windows
-  z/OS

et pour IBM MQ MQI clients connecté à ces systèmes.

2. Pris en charge sur les plateformes suivantes:

-  AIX
-  IBM i
-  Windows

et pour les clients IBM MQ connectés à ces systèmes.

3.  Pris en charge sur z/OS.

4.  Non pris en charge sous z/OS.

5. Non pris en charge sur VSE/ESA.

Sélecteur	Description	Remarque
CAALTD	Date de la dernière modification (LNDATE)	1
CAALTT	Heure de la dernière modification (LNTIME)	1
CALSTD	Description de la liste de noms (LNNLD)	1

Sélecteur	Description	Remarque
CALSTN	Nom de l'objet liste de noms (LNNLN)	1
CANAMS	Noms dans la liste de noms (LNQN x Nombre de noms dans la liste)	1
IANAMC	Nombre de noms dans la liste de noms	1
ID SGD	Disposition d'un groupe de partage de files d'attente	3

Sélecteur	Description	Remarque
CAALTD	Date de la dernière modification (LNDATE)	1
CAALTT	Heure de la dernière modification (LNTIME)	1
Interface CAAPPI	Identificateur d'application (LNPROA)	5
CAENVD	Données d'environnement (LNPROE)	5
CAPROD	Description de la définition de processus (LNPROD)	5
CAPRON	Nom de la définition de processus (LNPRON)	5
CAUSRD	Données utilisateur (LNPROU)	5
IAAPPT	Type d'application	5
ID SGD	Disposition d'un groupe de partage de files d'attente	3

Sélecteur	Description	Remarque
CAALTD	Date de la dernière modification (LNDATE)	1
CAALTT	Heure de la dernière modification (LNTIME)	1
CACADX	Nom d'exit de définition de canal automatique (LNEXN)	1
CACLWD	Données transmises à l'exit de charge de travail du cluster (LNEXDA)	1
CACLWX	Nom de l'exit de charge de travail de cluster (LNEXN)	1
CACMDQ	Nom de la file d'entrée des commandes système (LNQN)	5
File d'attente CADLQ	Nom de la file d'attente de rebut (LNQN)	5
Nom de file d'attente CADXQN	Nom de file d'attente de transmission par défaut (LNQN)	5
CAQMD	Description du gestionnaire de files d'attente (LNQMD)	5
ID AQMC	Identificateur de gestionnaire de files d'attente (LNQMID)	1
Réseau CMAQMN	Nom du gestionnaire de files d'attente local (LNQMN)	5
CAQSGN	Nom du groupe de partage de files d'attente (LNQSGN)	3

<i>Tableau 749. Sélecteurs d'attributs MQINQ pour le gestionnaire de files d'attente (suite)</i>		
<b>Sélecteur</b>	<b>Description</b>	<b>Remarque</b>
CARPN	Nom du cluster pour lequel le gestionnaire de files d'attente fournit des services de référentiel (LNQMN)	1
CARPNL	Nom de l'objet liste de noms contenant les noms des clusters pour lesquels le gestionnaire de files d'attente fournit des services de référentiel (LNNLN)	1
CMDEV	Attribut de contrôle qui détermine si les messages générés lors de l'émission de commandes sont placés dans une file d'attente	8
IAAUTE	Attribut de contrôle des événements de droits d'accès	4, 5
IACAD	Attribut de contrôle pour la définition de canal automatique	2
IACADE	Attribut de contrôle pour les événements de définition de canal automatique	2
IACLXQ	Type de file d'attente de transmission de cluster par défaut	4
IACLWL	Longueur de charge de travail de cluster	1
IACCSI	Identificateur de jeu de caractères codés	5
IACMDL	Niveau de commande pris en charge par le gestionnaire de files d'attente	5
IACFGE	Attribut de contrôle pour les événements de configuration	3
IADIST	Prise en charge de la liste de diffusion	2
IAINHE	Attribut de contrôle pour les événements d'interdiction	4, 5
CYCLE D'APPLICATIO N	Attribut de contrôle pour les événements locaux	4, 5
IAMHND	Nombre maximal de descripteurs	5
IAMLEN	Longueur maximale des messages	5
IAMPRI	Priorité maximum	5
IAMUNC	Nombre maximal de messages non validés dans une unité de travail	5
IAPFME	Attribut de contrôle pour les événements de performances	4, 5
IAPLAT	Plateforme sur laquelle réside le gestionnaire de files d'attente	5
IARMTE	Attribut de contrôle pour les événements distants	4, 5
IASSE	Attribut de contrôle pour les événements d'arrêt de démarrage	4, 5
IASYNC	Disponibilité du point de synchronisation	5
IATRLFT	Durée de vie des rubriques non administratives inutilisées	
IATRGI	Intervalle de déclenchement	5

#### **IACNT (entier signé à 10 chiffres)-entrée**

Nombre d'attributs de type entier.

Il s'agit du nombre d'éléments du tableau INTATR . Zéro est une valeur valide.

S'il s'agit au moins du nombre de sélecteurs IA\* dans le paramètre **SELS** , tous les attributs entiers demandés sont renvoyés.

### **INTATR (entier signé à 10 chiffres x IACNT)-sortie**

Tableau d'attributs de type entier.

Il s'agit d'un tableau de valeurs d'attribut d'entier *IACNT* .

Les valeurs d'attribut entières sont renvoyées dans le même ordre que les sélecteurs IA\* dans le paramètre **SELS** . Si le tableau contient plus d'éléments que le nombre de sélecteurs IA\*, les éléments en excès restent inchangés.

Si HOBJ représente une file d'attente, mais qu'un sélecteur d'attribut n'est pas applicable à ce type de file d'attente, la valeur spécifique IAVNA est renvoyée pour l'élément correspondant dans le tableau INTATR .

### **CALEN (entier signé à 10 chiffres)-entrée**

Longueur de la mémoire tampon des attributs de caractères.

Il s'agit de la longueur en octets du paramètre **CHRATR** .

Il doit s'agir au moins de la somme des longueurs des attributs de caractères demandés (voir SELS). Zéro est une valeur valide.

### **CHRATR (chaîne de caractères d'1 octet x CALEN)-sortie**

Attributs de caractères.

Il s'agit de la mémoire tampon dans laquelle les attributs de caractère sont renvoyés, concaténés. La longueur de la mémoire tampon est indiquée par le paramètre **CALEN** .

Les attributs de caractère sont renvoyés dans le même ordre que les sélecteurs CA\* dans le paramètre **SELS** . La longueur de chaque chaîne d'attribut est fixe pour chaque attribut (voir SELS) et la valeur qu'il contient est complétée à droite par des blancs si nécessaire. Si la mémoire tampon est supérieure à celle nécessaire pour contenir tous les attributs de caractères demandés (y compris le remplissage), les octets au-delà de la dernière valeur d'attribut renvoyée restent inchangés.

Si HOBJ représente une file d'attente, mais qu'un sélecteur d'attribut n'est pas applicable à ce type de file d'attente, une chaîne de caractères composée entièrement d'astérisques (\*) est renvoyée en tant que valeur de cet attribut dans CHRATR.

### **CMPCOD (entier signé à 10 chiffres)-sortie**

Code achèvement.

Il peut s'agir de :

#### **CCOK**

Achèvement réussi.

#### **CCWARN**

Avertissement (achèvement partiel).

#### **CCFAIL**

Echec de l'appel.

### **REASON (entier signé à 10 chiffres)-sortie**

Code anomalie qualifiant CMPCOD.

Si CMPCOD est CCOK:

#### **RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si CMPCOD est CCWARN:

#### **RC2008**

(2008, X'7D8') Espace insuffisant autorisé pour les attributs de caractères.

#### **RC2022**

(2022, X'7E6') Espace insuffisant autorisé pour les attributs de type entier.



**RC2068**

(2068, X'814') Sélecteur non applicable au type de file d'attente.

Si *CMPCOD* est CCFAIL:

**RC2219**

(2219, X'8AB') Appel MQI entré à nouveau avant la fin de l'appel précédent.

**RC2006**

(2006, X'7D6') Longueur des attributs de caractère non valide.

**RC2007**

(2007, X'7D7') Chaîne d'attributs de caractères non valide.

**RC2009**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**RC2018**

(2018, X'7E2') Descripteur de connexion non valide.

**RC2019**

(2019, X'7E3') Descripteur d'objet non valide.

**RC2021**

(2021, X'7E5') Nombre d'attributs de type entier non valide.

**RC2023**

(2023, X'7E7') Tableau d'attributs d'entier non valide.

**RC2038**

(2038, X'7F6') File d'attente non ouverte pour interrogation.

**RC2041**

(2041, X'7F9') Définition d'objet modifiée depuis son ouverture.

**RC2101**

(2101, X'835') Objet endommagé.

**RC2052**

(2052, X'804') File d'attente supprimée.

**RC2058**

(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

**RC2059**

(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

**RC2162**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**RC2102**

(2102, X'836') Ressources système disponibles insuffisantes.

**RC2065**

(2065, X'811') Nombre de sélecteurs non valide.

**RC2067**

(2067, X'813') Sélecteur d'attribut non valide.

**RC2066**

(2066, X'812') Nombre de sélecteurs trop grand.

**RC2071**

(2071, X'817') Mémoire disponible insuffisante.

**RC2195**

(2195, X'893') Une erreur inattendue s'est produite.

**Déclaration RPG**

C\*..1.....2.....3.....4.....5.....6.....7..

```

C          CALLP      MQINQ(HCONN : HOBJ : SELCNT :
C                      SELS(1) : IACNT : INTATR(1) :
C                      CALEN : CHRATR : CMPCOD :
C                      REASON)

```

La définition de prototype pour l'appel est la suivante:

```

D*.1.....2.....3.....4.....5.....6.....7..
DMQINQ          PR          EXTPROC('MQINQ')
D* Connection handle
D HCONN          10I 0 VALUE
D* Object handle
D HOBJ          10I 0 VALUE
D* Count of selectors
D SELCNT        10I 0 VALUE
D* Array of attribute selectors
D SELS          10I 0
D* Count of integer attributes
D IACNT          10I 0 VALUE
D* Array of integer attributes
D INTATR        10I 0
D* Length of character attributes buffer
D CALEN          10I 0 VALUE
D* Character attributes
D CHRATR          * VALUE
D* Completion code
D CMPCOD        10I 0
D* Reason code qualifying CMPCOD
D REASON        10I 0

```

IBM i

## MQINQMP (propriété de message d'interrogation) sur IBM i

L'appel MQINQMP renvoie la valeur d'une propriété d'un message.

- «Syntaxe», à la page 1378
- «Paramètres», à la page 1378
- «Déclaration RPG», à la page 1382

### Syntaxe

MQINQMP (*Hconn*, *Hmsg*, *InqPropOpts*, *Name*, *PropDesc*, *Type*, *ValueLength*, *Value*, *DataLength*, *CompCode*, *Reason*)

### Paramètres

L'appel MQINQMP comporte les paramètres suivants:

#### HCONN (entier signé à 10 chiffres)-entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* doit correspondre au descripteur de connexion qui a été utilisé pour créer le descripteur de message spécifié dans le paramètre **Hmsg**.

Si le descripteur de message a été créé à l'aide de HCUNAS, une connexion valide doit être établie sur l'unité d'exécution demandant une propriété du descripteur de message, sinon l'appel échoue avec RC2009.

#### HMSG (entier signé à 20 chiffres)-entrée

Il s'agit du descripteur de message à contrôler. La valeur a été renvoyée par un appel **MQCRTMH** précédent.

#### INQOPT (MQIMPO)-entrée

Pour plus de détails, voir le type de données MQIMPO.

## **PRNAME (MQCHARV)-entrée**

Décrit le nom de la propriété à interroger.

Si aucune propriété portant ce nom n'est trouvée, l'appel échoue avec la raison RC2471.

Vous pouvez utiliser le signe pourcentage (%) à la fin du nom de la propriété. Le caractère générique correspond à zéro ou plusieurs caractères, y compris le point (.). Cela permet à une application d'interroger la valeur de nombreuses propriétés. Appelez MQINQMP avec l'option IPINQF pour obtenir la première propriété correspondante et de nouveau avec l'option IPINQN pour obtenir la propriété correspondante suivante. Si aucune autre propriété correspondante n'est disponible, l'appel échoue avec RC2471. Si la zone *ReturnedName* de la structure InqPropOpts est initialisée avec une adresse ou un décalage pour le nom renvoyé de la propriété, cette opération est effectuée lors du retour de MQINQMP avec le nom de la propriété qui a été mise en correspondance. Si la zone *VSBufSize* de *ReturnedName* dans la structure InqPropOpts est inférieure à la longueur du nom de propriété renvoyé, le code achèvement est défini sur CCFAIL avec le motif RC2465.

Les propriétés qui possèdent des synonymes connus sont renvoyées comme suit:

1. Propriétés avec le préfixe "mqps." sont renvoyés avec le nom de propriété IBM MQ . Par exemple, "MQTopicString" est le nom renvoyé à la place de "mqps.Top".
2. Propriétés avec le préfixe "jms." ou "mcd." sont renvoyés en tant que nom de zone d'en-tête JMS . Par exemple, "JMSExpiration" est le nom renvoyé et non "jms.Exp".
3. Propriétés avec le préfixe "usr." sont renvoyés sans ce préfixe. Par exemple, "Color" est renvoyé à la place de "usr.Color".

Les propriétés avec des synonymes ne sont renvoyées qu'une seule fois.

Dans le langage de programmation RPG, les variables de macro suivantes sont définies pour l'interrogation de toutes les propriétés et de toutes les propriétés commençant par "usr.":

### **INQALL**

Consultez toutes les propriétés du message.

### **INQUSR**

Consultez toutes les propriétés du message qui démarrent "usr.". Le nom renvoyé est renvoyé sans la valeur "usr." .

Si IPINQN est indiqué mais que le nom a été modifié depuis l'appel précédent ou qu'il s'agit du premier appel, IPINQF est implicite.

Pour plus d'informations sur l'utilisation des noms de propriété, voir [Noms de propriété et Restrictions relatives aux noms de propriété](#) .

## **PRPDSC (MQPD)-sortie**

Cette structure est utilisée pour définir les attributs d'une propriété, y compris ce qui se produit si la propriété n'est pas prise en charge, le contexte de message auquel la propriété appartient et les messages dans lesquels la propriété doit être copiée. Pour plus d'informations sur cette structure, voir [MQPD](#) .

## **TYPE (entier signé à 10 chiffres)-entrée/sortie**

En cas de retour de l'appel MQINQMP, ce paramètre est défini sur le type de données *Valeur*. Le type de données peut être l'un des suivants:

### **Connaissance de type**

Une valeur booléenne.

### **TYPBST**

une chaîne d'octets.

### **TYPI8**

Entier signé de 8 bits.

### **TYPI16**

Entier signé 16 bits.

**TYPI32**

Entier signé 32 bits.

**TYPI64**

Entier signé 64 bits.

**TYPF32**

Nombre à virgule flottante 32 bits.

**TYPF64**

Nombre en virgule flottante 64 bits.

**TYPSTR**

Chaîne de caractères.

**TYPNUL**

La propriété existe mais a une valeur nulle.

Si le type de données de la valeur de propriété n'est pas reconnu, TYPSTR est renvoyé et une représentation sous forme de chaîne de la valeur est placée dans la zone *Valeur* . Une représentation sous forme de chaîne du type de données est disponible dans la zone *IPITYP* du paramètre *IPOPT* . Un code achèvement d'avertissement est renvoyé avec le motif RC2467.

De plus, si l'option IPCTYP est spécifiée, la conversion de la valeur de propriété est demandée. Utilisez *Type* comme entrée pour spécifier le type de données sous lequel vous souhaitez que la propriété soit renvoyée. Pour plus de détails sur la conversion de type de données, voir la description de l'option IPCTYP de «MQIMPO (Inquire message property options) sur IBM i», à la page 1155 .

Si vous ne demandez pas de conversion de type, vous pouvez utiliser la valeur suivante en entrée:

**TYPAST**

La valeur de la propriété est renvoyée sans convertir son type de données.

**VALLEN (entier signé à 10 chiffres)-entrée**

Longueur en octets de la zone *Valeur*.

Spécifiez zéro pour les propriétés pour lesquelles vous n'avez pas besoin de la valeur renvoyée. Il peut s'agir de propriétés conçues par une application pour avoir une valeur nulle ou une chaîne vide. Indiquez également zéro si l'option IPQLEN a été spécifiée ; dans ce cas, aucune valeur n'est renvoyée.

**VALUE (bit 1 octet stringxVALLEN)-sortie**

Il s'agit de la zone devant contenir la valeur de la propriété interrogée. La mémoire tampon doit être alignée sur une limite appropriée pour la valeur renvoyée. Si vous ne le faites pas, une erreur peut se produire lors de l'accès ultérieur à la valeur.

Si *VALLEN* est inférieur à la longueur de la valeur de la propriété, la plus grande partie de la valeur de la propriété est déplacée dans *VALUE* et l'appel échoue avec le code achèvement CCFAIL et la raison RC2469.

Le jeu de caractères des données dans *VALUE* est fourni par la zone IPRETCSI dans le paramètre INQOPT. Le codage des données dans *VALUE* est donné par la zone IPRETENC dans le paramètre INQOPT.

Si le paramètre *VALLEN* est à zéro, *VALUE* n'est pas référencé.

**DATLEN (entier signé à 10 chiffres)-sortie**

Il s'agit de la longueur en octets de la valeur de propriété réelle renvoyée dans la zone *Valeur* .

Si *DataLength* est inférieur à la longueur de la valeur de la propriété, *DataLength* est toujours entré lors du retour de l'appel MQINQMP. Cela permet à l'application de déterminer la taille de la mémoire tampon requise pour la valeur de propriété, puis d'émettre à nouveau l'appel avec une mémoire tampon de la taille appropriée.

Les valeurs suivantes peuvent également être renvoyées.

Si le paramètre *Type* est défini sur TYPSTR ou TYPBST:

**VLEMP**

La propriété existe mais ne contient pas de caractères ni d'octets.

**CMPCOD (entier signé à 10 chiffres)-sortie**

Code achèvement ; il s'agit d'un des codes suivants :

**CCOK**

Achèvement réussi.

**CCWARN**

Avertissement (achèvement partiel).

**CCFAIL**

Echec de l'appel.

**REASON (entier signé à 10 chiffres)-sortie**

Code anomalie qualifiant *CompCode*.

Si *CMPCOD* est CCOK:

**RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est CCWARN:

**RC2492**

(2492, X'09BC') Nom de propriété renvoyé non converti.

**RC2466**

(2466, X'09A2') Valeur de propriété non convertie.

**RC2467**

(2467, X'09A3') Le type de données de propriété n'est pas pris en charge.

**RC2421**

(2421, X'0975') Un dossier MQRFH2 contenant des propriétés n'a pas pu être analysé.

Si *CMPCOD* est CCFAIL:

**RC2204**

(2204, X'089C') Carte non disponible.

**RC2130**

(2130, X'0852') Impossible de charger le module de service de l'adaptateur.

**RC2157**

(2157, X'086D') Les ASID principal et principal diffèrent.

**RC2004**

(2004, X'07D4') Paramètre de valeur non valide.

**RC2005**

(2005, X'07D5') Paramètre de longueur de valeur non valide.

**RC2219**

(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

**RC2009**

(2009, X'07D9') La connexion au gestionnaire de files d'attente a été perdue.

**RC2010**

(2010, X'07DA') Paramètre de longueur de données non valide.

**RC2464**

(2464, X'09A0') La structure des options de propriété de message d'interrogation n'est pas valide.

**RC2460**

(2460, X'099C') Descripteur de message non valide.

**RC2499**

(2499, X'09C3') Descripteur de message déjà utilisé.

**RC2064**

(2046, X'07F8') Options non valides ou non cohérentes.

**RC2482**

(2482, X'09B2') Structure de descripteur de propriété non valide.

**RC2470**

(2470, X'09A6') La conversion du type de données réel au type de données demandé n'est pas prise en charge.

**RC2442**

(2442, X'098A') Nom de propriété non valide.

**RC2465**

(2465, X'09A1') Nom de propriété trop grand pour la mémoire tampon de nom renvoyée.

**RC2471**

(2471, X'09A7) Propriété non disponible.

**RC2469**

(2469, X'09A5') Valeur de propriété trop grande pour la zone Valeur.

**RC2472**

(2472, X'09A8') Erreur de format numérique détectée dans les données de valeur.

**RC2473**

(2473, X'09A9') Type de propriété demandé non valide.

**RC2111**

(2111, X'083F') Identificateur de jeu de caractères codés de nom de propriété non valide.

**RC2071**

(2071, X'0871') Mémoire disponible insuffisante.

**RC2195**

(2195, X'0893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir:

- [Messages, codes achèvement et codes anomalie IBM MQ for z/OS pour IBM MQ for z/OS](#)
- [Messages et codes anomalie](#) pour toutes les autres plateformes IBM MQ

## Déclaration RPG

```

C*.1.....2.....3.....4.....5.....6.....7..
C          CALLP          MQINQMP(HCONN : HMSG : INQOPT :
                          PRNAME : PRPDSC : TYPE :
                          VALLEN : VALUE : DATLEN :
                          CMPCOD : REASON)

```

La définition de prototype pour l'appel est la suivante:

```

DMQINQMP          PR          EXTPROC('MQINQMP')
D* Connection handle
D HCONN          10I 0 VALUE
D* Message handle
D HMSG          20I 0 VALUE
D* Options that control the action of MQINQMP
D INQOPT          72A
D* Property name
D PRNAME          32A
D* Property descriptor
D PRPDSC          24A
D* Property data type
D TYPE          10I 0
D* Length in bytes of the Value area
D VALLEN          10I 0 VALUE
D* Property value

```

D VALUE	*	VALUE
D* Length of the property value		
D DATLEN		10I 0
D* Completion code		
D CMPCOD		10I 0
D* Reason code qualifying CompCode		
D REASON		10I 0

## MQMHBUF (conversion de l'identificateur de message en mémoire tampon) sous IBM i

Le MQMHBUF convertit un descripteur de message en mémoire tampon et est l'inverse de l'appel MQBUFMH.

- «Syntaxe», à la page [1383](#)
- «Remarques sur l'utilisation», à la page [1383](#)
- «Paramètres», à la page [1383](#)
- «Déclaration RPG», à la page [1385](#)

### Syntaxe

MQMHBUF (*Hconn*, *Hmsg*, *MsgHBufOpts*, *Name*, *MsgDesc*, *BufferLength*, *Buffer*, *DataLength*, *CompCode*, *Reason*)

### Remarques sur l'utilisation

MQMHBUF convertit un descripteur de message en mémoire tampon.

Vous pouvez l'utiliser avec un exit d'API MQGET pour accéder à certaines propriétés, à l'aide des API de propriété de message, puis transmettre ces propriétés dans une mémoire tampon à une application conçue pour utiliser les en-têtes MQRFH2 plutôt que les descripteurs de message.

Cet appel est l'inverse de l'appel MQBUFMH, que vous pouvez utiliser pour analyser les propriétés de message d'une mémoire tampon dans un descripteur de message.

### Paramètres

L'appel MQMHBUF comporte les paramètres suivants:

#### HCONN (entier signé à 10 chiffres)-entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente.

La valeur de *HCONN* doit correspondre au descripteur de connexion qui a été utilisé pour créer le descripteur de message spécifié dans le paramètre **HMSG**.

Si le descripteur de message a été créé à l'aide de HCUNAS, une connexion valide doit être établie sur l'unité d'exécution qui supprime le descripteur de message. Si une connexion valide n'est pas établie, l'appel échoue avec RC2009.

#### HMSG (entier signé à 20 chiffres)-entrée

Ce descripteur est le descripteur de message pour lequel une mémoire tampon est requise.

La valeur a été renvoyée par un appel MQCRTMH précédent.

#### MHBOPT (MQMHBO)-entrée

La structure MQMHBO permet aux applications de spécifier des options qui contrôlent la façon dont les mémoires tampon sont produites à partir des descripteurs de message.

Pour plus d'informations, voir «MQBMHO (Buffer to message handle options) sur IBM i», à la page [1065](#).

### **PRNAME (MQCHARV)-entrée**

Nom de la ou des propriétés à placer dans la mémoire tampon.

Si aucune propriété correspondant au nom n'est trouvée, l'appel échoue avec RC2471.

#### **Caractères génériques**

Vous pouvez utiliser un caractère générique pour placer plusieurs propriétés dans la mémoire tampon. Pour ce faire, utilisez le signe de pourcentage (%) à la fin du nom de la propriété. Ce caractère générique correspond à zéro ou plusieurs caractères, y compris le caractère point (.).

Pour plus d'informations sur l'utilisation des noms de propriété, voir [Noms de propriété et Restrictions relatives aux noms de propriété](#).

### **MSGDSC (MQMD)-entrée / sortie**

La structure *MSGDSC* décrit le contenu de la zone tampon.

Dans la sortie, les zones *Encoding*, *CodedCharSetId* et *Format* sont définies pour décrire correctement le codage, l'identificateur de jeu de caractères et le format des données dans la zone tampon, tels qu'ils sont écrits par l'appel.

Les données de cette structure se trouvent dans le jeu de caractères et le codage de l'application.

### **BUFLEN (entier signé à 10 chiffres)-entrée**

*BUFLEN* est la longueur de la zone tampon, en octets.

### **BUFFER (chaîne de 1 octet x BUFLEN)-entrée / sortie**

*BUFFER* définit la zone contenant la mémoire tampon de messages. Pour la plupart des données, vous devez aligner la mémoire tampon sur une limite de 4 octets.

Si *BUFFER* contient des données de type caractère ou numérique, définissez les zones *CodedCharSetId* et *Encoding* du paramètre **MSGDSC** sur les valeurs appropriées aux données ; cela permet la conversion des données, si nécessaire.

Si des propriétés sont trouvées dans la mémoire tampon de messages, elles sont éventuellement supprimées ; elles sont ensuite disponibles à partir du descripteur de message lors du retour de l'appel.

Dans le langage de programmation C, le paramètre est déclaré comme un pointeur vers vide, ce qui signifie que l'adresse de tout type de données peut être spécifiée comme paramètre.

Si le paramètre **BUFLEN** a pour valeur zéro, il n'est pas fait référence à *BUFFER*. Dans ce cas, l'adresse de paramètre transmise par les programmes écrits en assembleur C ou System/390 peut être null.

### **DATLEN (entier signé à 10 chiffres)-sortie**

*DATLEN* est la longueur, en octets, des propriétés renvoyées dans la mémoire tampon. Si la valeur est zéro, aucune propriété ne correspond à la valeur indiquée dans *PRNAME* et l'appel échoue avec le code anomalie RC2471.

Si *BUFLEN* est inférieur à la longueur requise pour stocker les propriétés dans la mémoire tampon, l'appel MQMHBUFF échoue avec RC2469, mais une valeur est toujours entrée dans *DATLEN*. Cela permet à l'application de déterminer la taille de la mémoire tampon requise pour les propriétés, puis d'émettre à nouveau l'appel avec le *BUFLEN* requis.

### **CMPCOD (entier signé à 10 chiffres)-sortie**

Code achèvement ; il s'agit d'un des codes suivants :

#### **CCOK**

Achèvement réussi.

#### **CCFAIL**

Echec de l'appel.



## REASON (entier signé à 10 chiffres)-sortie

Code anomalie qualifiant *CMPCOD*.

Si *CMPCOD* est CCOK:

### RCNONE

(0, X'000') Aucun code anomalie à signaler.

Si *CMPCOD* est CCFAIL:

### RC2204

(2204, X'089C') Carte non disponible.

### RC2130

(2130, X'852') Impossible de charger le module de service d'adaptateur.

### RC2157

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

### RC2501

(2501, X'095C') Le descripteur de message vers la structure des options de mémoire tampon n'est pas valide.

### RC2004

(2004, X'07D4') Paramètre de mémoire tampon non valide.

### RC2005

(2005, X'07D5') Paramètre de longueur de mémoire tampon non valide.

### RC2219

(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

### RC2009

(2009, X'07D9') La connexion au gestionnaire de files d'attente a été perdue.

### RC2010

(2010, X'07DA') Paramètre de longueur de données non valide.

### RC2460

(2460, X'099C') Descripteur de message non valide.

### RC2026

(2026, X'07EA') Descripteur de message non valide.

### RC2499

(2499, X'09C3') Descripteur de message déjà utilisé.

### RC2046

(2046, X'07FE') Options non valides ou non cohérentes.

### RC2442

(2442, X'098A') Le nom de propriété n'est pas valide.

### RC2471

(2471, X'09A7') Propriété non disponible.

### RC2469

(2469, X'09A5') La valeur BufferLength est trop petite pour contenir les propriétés spécifiées.

### RC2195

(2195, X'893') Une erreur inattendue s'est produite.

## Déclaration RPG

```
C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQMHBUF(HCONN : HMSG : MHBOPT :
                        PRNAME : MSGDSC : BUFLen :
                        BUFFER : DATLEN :
                        CMPCOD : REASON)
```

La définition de prototype pour l'appel est la suivante:

```
DMQMHBUFF          PR          EXTPROC('MQMHBUFF')
D* Connection handle
D HCONN             10I 0 VALUE
D* Message handle
D HMSG              20I 0 VALUE
D* Options that control the action of MQMHBUFF
D MHBOP             12A
D* Property name
D PRNAME            32A
D* Message descriptor
D MSGDSC            364A
D* Length in bytes of the Buffer area
D BUFLLEN           10I 0 VALUE
D* Area to contain the properties
D BUFFER            *   VALUE
D* Length of the properties
D DATLEN            10I 0
D* Completion code
D CMPCOD            10I 0
D* Reason code qualifying CompCode
D REASON            10I 0
```

## IBM i MQOPEN (objet ouvert) sur IBM i

L'appel MQOPEN établit l'accès à un objet.

Les types d'objet suivants sont valides:

- File d'attente (y compris les listes de distribution)
- Liste de noms
- Définition de processus
- Gestionnaire de files d'attente
- Topic

### Index

- [«Syntaxe», à la page 1386](#)
- [«Remarques sur l'utilisation», à la page 1386](#)
- [«Paramètres», à la page 1391](#)
- [«Déclaration RPG», à la page 1398](#)

### Syntaxe

MQOPEN (*HCONN*, *OBJDSC*, *OPTS*, *HOBJ*, *CMPCOD*, *REASON*)

### Remarques sur l'utilisation

1. L'objet ouvert est l'un des suivants:

- Une file d'attente, afin de:
  - Extraire ou parcourir les messages (à l'aide de l'appel MQGET)
  - Messages d'insertion (à l'aide de l'appel MQPUT)
  - Renseignez-vous sur les attributs de la file d'attente (à l'aide de l'appel MQINQ)
  - Définir les attributs de la file d'attente (à l'aide de l'appel MQSET)

Si la file d'attente nommée est une file d'attente modèle, une file d'attente locale dynamique est créée.

Une liste de distribution est un type spécial d'objet file d'attente qui contient une liste de files d'attente. Il peut être ouvert pour insérer des messages, mais pas pour extraire ou parcourir des messages, ni pour interroger ou définir des attributs. Voir la note d'utilisation 8 pour plus de détails.

Une file d'attente QSGDISP (GROUP) est un type spécial de définition de file d'attente qui ne peut pas être utilisé avec les appels MQOPEN ou MQPUT1 .

- Une liste de noms, afin de:
    - Renseignez-vous sur les noms des files d'attente de la liste (à l'aide de l'appel MQINQ).
  - Une définition de processus, afin de:
    - Renseignez-vous sur les attributs de processus (à l'aide de l'appel MQINQ).
  - Le gestionnaire de files d'attente, afin de:
    - Renseignez-vous sur les attributs du gestionnaire de files d'attente local (à l'aide de l'appel MQINQ).
2. Une application peut ouvrir le même objet plusieurs fois. Un descripteur d'objet différent est renvoyé pour chaque ouverture. Chaque descripteur renvoyé peut être utilisé pour les fonctions pour lesquelles l'ouverture correspondante a été effectuée.
3. Si l'objet en cours d'ouverture est une file d'attente mais pas une file d'attente de cluster, la résolution de tous les noms dans le gestionnaire de files d'attente local a lieu au moment de l'appel MQOPEN. Cela peut inclure un ou plusieurs des éléments suivants pour un appel MQOPEN particulier:
- Résolution d'alias sur le nom d'une file d'attente de base
  - Résolution du nom d'une définition locale d'une file d'attente éloignée en nom du gestionnaire de files d'attente éloignées et nom sous lequel la file d'attente est connue sur le gestionnaire de files d'attente éloignées
  - Résolution du nom du gestionnaire de files d'attente éloignées en nom de file d'attente de transmission locale

Cependant, sachez que les appels MQINQ ou MQSET ultérieurs pour le descripteur se rapportent uniquement au nom qui a été ouvert et non à l'objet résultant de la résolution de nom. Par exemple, si l'objet ouvert est un alias, les attributs renvoyés par l'appel MQINQ sont les attributs de l'alias et non les attributs de la file d'attente de base dans laquelle l'alias est résolu. La vérification de la résolution de nom est toujours effectuée, indépendamment de ce qui est indiqué pour le paramètre **OPTS** sur le MQOPEN correspondant.

Si l'objet en cours d'ouverture est une file d'attente de cluster, la résolution de nom peut se produire au moment de l'appel MQOPEN ou être différée jusqu'à une date ultérieure. Le point de résolution est contrôlé par les options OOBND\* spécifiées dans l'appel MQOPEN:

- OOBND0
- Nom distinctif OOBNDN
- OOBNDQ

Voir [Résolution de nom](#) pour plus d'informations sur la résolution de nom pour les files d'attente de cluster.

4. Les attributs d'un objet peuvent changer lorsque l'objet est ouvert dans une application. Dans de nombreux cas, l'application ne le remarque pas, mais pour certains attributs, le gestionnaire de files d'attente marque l'identificateur comme n'étant plus valide. Il s'agit des fonctions suivantes :
- Tout attribut qui affecte la résolution de nom de l'objet. Cela s'applique indépendamment des options ouvertes utilisées et inclut les éléments suivants:
    - Modification de l'attribut **BaseQName** d'une file d'attente alias ouverte.
    - Modification des attributs de file d'attente **RemoteQName** ou **RemoteQMgrName** , pour tout descripteur ouvert pour cette file d'attente ou pour une file d'attente qui se résout via cette définition en tant qu'alias de gestionnaire de files d'attente.

- Toute modification entraînant la résolution d'un descripteur actuellement ouvert pour une file d'attente éloignée en une autre file d'attente de *transmission* ou l'échec de la résolution en une seule file d'attente. Par exemple, il peut s'agir des éléments suivants:
  - Modification de l'attribut **XmitQName** de la définition locale d'une file d'attente éloignée, que la définition soit utilisée pour une file d'attente ou pour un alias de gestionnaire de files d'attente.
 

Il y a une exception à cela, à savoir la création d'une nouvelle file d'attente de transmission. Un descripteur qui aurait été résolu dans cette file d'attente s'il avait été présent lors de l'ouverture du descripteur, mais qui aurait été résolu dans la file d'attente de transmission par défaut, n'est pas rendu non valide.
  - Modification de l'attribut de gestionnaire de files d'attente **DefXmitQName** . Dans ce cas, tous les descripteurs ouverts résolus dans la file d'attente précédemment nommée (qui ont été résolus uniquement parce qu'il s'agissait de la file d'attente de transmission par défaut) sont marqués comme non valides. Les descripteurs résolus dans cette file d'attente pour d'autres raisons ne sont pas affectés.
- L'attribut de file d'attente **Shareability** , s'il existe deux descripteurs ou plus qui fournissent actuellement un accès OOINPS pour cette file d'attente ou pour une file d'attente qui se résout en cette file d'attente. Si tel est le cas, *tous* les descripteurs qui sont ouverts pour cette file d'attente, ou pour une file d'attente qui se résout en cette file d'attente, sont marqués comme non valides, quelles que soient les options d'ouverture.
- L'attribut de file d'attente **Usage** , pour tous les descripteurs ouverts pour cette file d'attente, ou pour une file d'attente qui se résout en cette file d'attente, quelles que soient les options d'ouverture.

Lorsqu'un descripteur est marqué comme non valide, tous les appels ultérieurs (autres que MQCLOSE) utilisant ce descripteur échouent avec le code anomalie RC2041; l'application doit émettre un appel MQCLOSE (à l'aide du descripteur d'origine), puis rouvrir la file d'attente. Toutes les mises à jour non validées par rapport à l'ancien descripteur des appels réussis précédents peuvent toujours être validées ou annulées, comme requis par la logique de l'application.

Si la modification d'un attribut entraîne cette situation, une version "force" spéciale de la commande doit être utilisée.

5. Le gestionnaire de files d'attente effectue des contrôles de sécurité lorsqu'un appel MQOPEN est émis, afin de vérifier que l'ID utilisateur sous lequel l'application s'exécute dispose du niveau de droits approprié avant que l'accès ne soit autorisé. La vérification des droits est effectuée sur le nom de l'objet en cours d'ouverture et non sur le ou les noms, ce qui se produit après la résolution d'un nom.

Si l'objet ouvert est une file d'attente modèle, le gestionnaire de files d'attente effectue une vérification de sécurité complète par rapport au nom de la file d'attente modèle et au nom de la file d'attente dynamique créée. Si la file d'attente dynamique résultante est ouverte de manière explicite, une vérification supplémentaire de la sécurité des ressources est effectuée par rapport au nom de la file d'attente dynamique.

6. Une file d'attente éloignée peut être spécifiée de l'une des deux manières suivantes dans le paramètre **OBJDSC** de cet appel (voir les zones *ODON* et *ODMN* décrites dans [«MQOD \(descripteur d'objet\) sous IBM i»](#), à la page 1215 ):
  - En spécifiant pour *ODON* le nom d'une définition locale de la file d'attente éloignée. Dans ce cas, *ODMN* fait référence au gestionnaire de files d'attente local et peut être vide.
 

La validation de sécurité effectuée par le gestionnaire de files d'attente local vérifie que l'utilisateur est autorisé à ouvrir la définition locale de la file d'attente éloignée.
  - En spécifiant pour *ODON* le nom de la file d'attente éloignée, tel qu'il est connu du gestionnaire de files d'attente éloignées. Dans ce cas, *ODMN* est le nom du gestionnaire de files d'attente éloignées.
 

La validation de sécurité effectuée par le gestionnaire de files d'attente local vérifie que l'utilisateur est autorisé à envoyer des messages à la file d'attente de transmission résultant du processus de résolution de nom.

Dans les deux cas:

- Aucun message n'est envoyé par le gestionnaire de files d'attente local au gestionnaire de files d'attente éloignées afin de vérifier que l'utilisateur est autorisé à placer des messages dans la file d'attente.
  - Lorsqu'un message arrive sur le gestionnaire de files d'attente éloignées, ce dernier peut le rejeter car l'utilisateur à l'origine du message n'est pas autorisé.
7. Un appel MQOPEN avec l'option OOBROW établit un curseur de navigation, à utiliser avec des appels MQGET qui spécifient le descripteur d'objet et l'une des options de navigation. Cela permet d'analyser la file d'attente sans modifier son contenu. Un message qui a été trouvé lors de la consultation peut être supprimé ultérieurement de la file d'attente à l'aide de l'option GMMUC.

Plusieurs curseurs de navigation peuvent être actifs pour une seule application en émettant plusieurs demandes MQOPEN pour la même file d'attente.

8. Les remarques suivantes s'appliquent à l'utilisation des listes de distribution.

- Les zones de la structure MQOD doivent être définies comme suit lors de l'ouverture d'une liste de distribution:
  - *ODVER* doit être *ODVER2* ou supérieur.
  - *ODOT* doit être *OTQ*.
  - *ODON* doit être vide ou la chaîne null.
  - *ODMN* doit être vide ou la chaîne null.
  - *ODREC* être supérieur à zéro.
  - L'un des éléments *ODORO* et *ODORP* doit être égal à zéro et l'autre doit être différent de zéro.
  - *ODRRO* et *ODRRP* ne peuvent pas être différents de zéro.
  - Il doit y avoir des enregistrements d'objet *ODREC*, traités par *ODORO* ou *ODORP*. Les enregistrements d'objet doivent être définis sur les noms des files d'attente de destination à ouvrir.
  - Si l'un des éléments *ODRRO* et *ODRRP* est différent de zéro, des enregistrements de réponse *ODREC* doivent être présents. Ils sont définis par le gestionnaire de files d'attente si l'appel se termine avec le code anomalie RC2136.

Un MQOD version-2 peut également être utilisé pour ouvrir une file d'attente unique qui ne figure pas dans une liste de distribution, en veillant à ce que *ODREC* soit égal à zéro.

- Seules les options d'ouverture suivantes sont valides dans le paramètre **OPTS** :
  - OOOUT
  - OOPAS\*
  - OOSET\*
  - OOALTU
  - OOFIQ
- Les files d'attente de destination de la liste de distribution peuvent être des files d'attente locales, des files d'attente alias ou des files d'attente éloignées, mais elles ne peuvent pas être des files d'attente modèles. Si une file d'attente modèle est spécifiée, l'ouverture de cette file d'attente échoue, avec le code anomalie RC2057. Toutefois, cela n'empêche pas l'ouverture d'autres files d'attente de la liste.
- Les paramètres de code achèvement et de code anomalie sont définis comme suit:
  - Si les opérations d'ouverture des files d'attente de la liste de distribution aboutissent ou échouent de la même manière, les paramètres de code achèvement et de code anomalie sont définis pour décrire le résultat commun. Les enregistrements de réponse MQRR (s'ils sont fournis par l'application) ne sont pas définis dans ce cas.

Par exemple, si chaque ouverture aboutit, le code achèvement est défini sur CCOK et le code anomalie est RCNONE ; si chaque ouverture échoue car aucune des files d'attente n'existe, les paramètres sont définis sur CCFAIL et RC2085.

- Si les opérations d'ouverture des files d'attente de la liste de distribution n'aboutissent pas toutes ou échouent de la même manière:
  - Le paramètre de code achèvement est défini sur CCWARN si au moins une ouverture a abouti et sur CCFAIL si toutes les opérations ont échoué.
  - Le paramètre de code anomalie est défini sur RC2136.
  - Les enregistrements de réponse (s'ils sont fournis par l'application) sont définis sur les codes achèvement individuels et les codes raison des files d'attente de la liste de distribution.

- Lorsqu'une liste de distribution a été ouverte avec succès, le descripteur *HOBJ* renvoyé par l'appel peut être utilisé sur les appels MQPUT suivants pour placer des messages dans les files d'attente de la liste de distribution et sur un appel MQCLOSE pour abandonner l'accès à la liste de distribution. La seule option de fermeture admise pour une liste de distribution est CONONE.

L'appel MQPUT1 peut également être utilisé pour placer un message dans une liste de distribution ; la structure MQOD définissant les files d'attente dans la liste est spécifiée en tant que paramètre sur cet appel.

- Chaque destination correctement ouverte dans la liste de distribution compte comme un descripteur *distinct* lorsque vous vérifiez si l'application a dépassé le nombre maximal de descripteurs autorisé (voir l'attribut de gestionnaire de files d'attente **MaxHandles** ). Cela est vrai même lorsque deux ou plusieurs des destinations de la liste de distribution sont effectivement résolues dans la même file d'attente physique. Si l'appel MQOPEN ou MQPUT1 d'une liste de distribution entraîne le dépassement du nombre de descripteurs utilisés par l'application *MaxHandles*, l'appel échoue avec le code anomalie RC2017.
- Chaque destination ouverte avec succès a la valeur de son attribut **OpenOutputCount** incrémentée d'un. Si au moins deux des destinations de la liste de distribution sont effectivement résolues dans la même file d'attente physique, l'attribut **OpenOutputCount** de cette file d'attente est incrémenté du nombre de destinations de la liste de distribution qui sont résolues dans cette file d'attente.
- Toute modification apportée aux définitions de file d'attente qui aurait entraîné la non-validité d'un descripteur si les files d'attente avaient été ouvertes individuellement (par exemple, une modification du chemin de résolution) n'entraîne pas la non-validité du descripteur de liste de distribution. Toutefois, elle entraîne un échec pour cette file d'attente particulière lorsque le descripteur de liste de distribution est utilisé lors d'un appel MQPUT ultérieur.
- Une liste de distribution ne peut contenir qu'une seule destination.

#### 9. Les remarques suivantes s'appliquent à l'utilisation des files d'attente de cluster.

- Lorsqu'une file d'attente de cluster est ouverte pour la première fois et que le gestionnaire de files d'attente local n'est pas un gestionnaire de files d'attente de référentiel complet, le gestionnaire de files d'attente local obtient des informations sur la file d'attente de cluster à partir d'un gestionnaire de files d'attente de référentiel complet. Lorsque le réseau est occupé, le gestionnaire de files d'attente local peut mettre plusieurs secondes à recevoir les informations requises du gestionnaire de files d'attente de référentiel. Par conséquent, l'application qui émet l'appel MQOPEN peut avoir à attendre jusqu'à 10 secondes avant que le contrôle ne soit renvoyé à partir de l'appel MQOPEN. Si le gestionnaire de files d'attente local ne reçoit pas les informations nécessaires sur la file d'attente de cluster dans ce délai, l'appel échoue avec le code anomalie RC2189.
- Lorsqu'une file d'attente de cluster est ouverte et qu'il existe plusieurs instances de la file d'attente dans le cluster, l'instance réellement ouverte dépend des options spécifiées dans l'appel MQOPEN:
  - Si les options spécifiées incluent l'une des options suivantes:
    - OOBRW
    - OOINPQ
    - OOINPX

- OOINPS
- OOSSET

l'instance de la file d'attente de cluster ouverte doit être l'instance locale. S'il n'existe aucune instance locale de la file d'attente, l'appel MQOPEN échoue.

- Si les options spécifiées n'incluent aucune des options ci-dessus, mais incluent l'une des options suivantes ou les deux:

- OOINQ
- OOOOT

l'instance ouverte est l'instance locale, le cas échéant, et une instance distante, dans le cas contraire. L'instance choisie par le gestionnaire de files d'attente peut toutefois être modifiée par un exit de charge de travail de cluster (s'il en existe un).

Pour plus d'informations sur les files d'attente de cluster, voir [Files d'attente de cluster](#).

10. Les applications démarrées par un moniteur de déclenchement sont transmises avec le nom de la file d'attente associée à l'application lors de son démarrage. Ce nom de file d'attente peut être spécifié dans le paramètre **OBJDSC** pour ouvrir la file d'attente. Pour plus de détails, voir la description de la structure MQTMC.
11. Lors de l'utilisation de l'option OORLOQ, la file d'attente locale est déjà renvoyée lorsqu'une file d'attente locale, alias ou modèle est ouverte, mais ce n'est pas le cas lorsque, par exemple, une file d'attente éloignée ou une file d'attente de cluster non locale est ouverte ; les noms ResolvedQName et ResolvedQMgrsont entrés avec les noms RemoteQName et RemoteQMgrtrouvés dans la définition de file d'attente éloignée, ou de la même manière avec la file d'attente de cluster éloignée choisie. Si OORLOQ est spécifié lors de l'ouverture, par exemple, d'une file d'attente éloignée, ResolvedQName sera désormais la file d'attente de transmission dans laquelle les messages seront insérés. Le nom ResolvedQMgrsera entré avec le nom du gestionnaire de files d'attente local qui héberge la file d'attente de transmission. Si un utilisateur est autorisé à parcourir, à entrer ou à sortir dans une file d'attente, il dispose des droits requis pour spécifier cet indicateur dans l'appel MQOPEN. Aucun droit spécial n'est requis.

## Paramètres

L'appel MQOPEN comporte les paramètres suivants:

### **HCONN (entier signé à 10 chiffres)-entrée**

Descripteur de connexion.

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *HCONN* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

### **OBJDSC (MQOD)-entrées-sorties**

Descripteur d'objet.

Il s'agit d'une structure qui identifie l'objet à ouvrir. Pour plus de détails, voir [«MQOD \(descripteur d'objet\) sous IBM i»](#), à la page 1215 .

Si la zone *ODON* du paramètre **OBJDSC** correspond au nom d'une file d'attente modèle, une file d'attente locale dynamique est créé avec les attributs de la file d'attente modèle ; cela se produit indépendamment des options d'ouverture spécifiées par le paramètre **OPTS** . Les opérations suivantes utilisant le *HOBJ* renvoyé par l'appel MQOPEN sont effectuées sur la nouvelle file d'attente dynamique et non sur la file d'attente modèle. Cela est vrai même pour les appels MQINQ et MQSET. Le nom de la file d'attente modèle dans le paramètre **OBJDSC** est remplacé par le nom de la file d'attente dynamique créée. Le type de la file d'attente dynamique est déterminé par la valeur de l'attribut **DefinitionType** de la file d'attente modèle (voir [«Attributs des files d'attente»](#), à la page 1435 ). Pour plus d'informations sur les options de fermeture applicables aux files d'attente dynamiques, voir la description de l'appel MQCLOSE.

## OPTS (entier signé à 10 chiffres)-entrée

Options qui contrôlent l'action de MQOPEN.

Au moins une des options suivantes doit être spécifiée:

- OOBW
- OOINP\* (un seul de ces éléments)
- OOINQ
- OOOU
- OOSET
- OORLQ

D'autres options peuvent être spécifiées selon les besoins. Si plusieurs options sont requises, les valeurs peuvent être ajoutées (n'ajoutez pas la même constante plusieurs fois). Les combinaisons qui ne sont pas valides sont notées ; toutes les autres combinaisons sont valides. Seules les options applicables au type d'objet indiqué par *OBJDSC* sont autorisées (voir [Options MQOPEN valides pour chaque type de file d'attente](#)).

**Options d'accès:** Les options suivantes contrôlent le type d'opérations pouvant être effectuées sur l'objet:

### OOINPQ

Ouvrez la file d'attente pour obtenir les messages en utilisant la valeur par défaut définie par la file d'attente.

La file d'attente est ouverte pour être utilisée avec les appels MQGET suivants. Le type d'accès est partagé ou exclusif, en fonction de la valeur de l'attribut de file d'attente **DefInputOpenOption** ; voir «Attributs des files d'attente», à la page 1435 pour plus de détails.

Cette option est valide uniquement pour les files d'attente locales, d'alias et modèles ; elle n'est pas valide pour les files d'attente éloignées, les listes de distribution et les objets qui ne sont pas des files d'attente.

### OOINPS

Ouvrez la file d'attente pour obtenir les messages avec accès partagé.

La file d'attente est ouverte pour être utilisée avec les appels MQGET suivants. L'appel peut aboutir si la file d'attente est actuellement ouverte par cette application ou par une autre application avec OOIINPS, mais échoue avec le code anomalie RC2042 si la file d'attente est actuellement ouverte avec OOIINPX.

Cette option est valide uniquement pour les files d'attente locales, d'alias et modèles ; elle n'est pas valide pour les files d'attente éloignées, les listes de distribution et les objets qui ne sont pas des files d'attente.

### OOINPX

Ouvrez la file d'attente pour obtenir les messages avec un accès exclusif.

La file d'attente est ouverte pour être utilisée avec les appels MQGET suivants. L'appel échoue avec le code anomalie RC2042 si la file d'attente est actuellement ouverte par cette application ou par une autre pour l'entrée de tout type (OOIINPS ou OOIINPX).

Cette option est valide uniquement pour les files d'attente locales, d'alias et modèles ; elle n'est pas valide pour les files d'attente éloignées, les listes de distribution et les objets qui ne sont pas des files d'attente.

Les remarques suivantes s'appliquent à ces options:

- Une seule de ces options peut être spécifiée.
- Un appel MQOPEN avec l'une de ces options peut aboutir même si l'attribut de file d'attente **InhibitGet** est défini sur QAGETI (bien que les appels MQGET suivants échouent alors que l'attribut est défini sur cette valeur).



- Si la file d'attente est définie comme n'étant pas partageable (c'est-à-dire que l'attribut de file d'attente **Shareability** a la valeur QANSHR), les tentatives d'ouverture de la file d'attente pour un accès partagé sont traitées comme des tentatives d'ouverture de la file d'attente avec un accès exclusif.
- Si une file d'attente alias est ouverte avec l'une de ces options, le test d'utilisation exclusive (ou de l'utilisation exclusive d'une autre application) est effectué sur la file d'attente de base dans laquelle l'alias est résolu.
- Ces options ne sont pas valides si *ODMN* est le nom d'un alias de gestionnaire de files d'attente ; cela est vrai même si la valeur de l'attribut **RemoteQMgrName** dans la définition locale d'une file d'attente éloignée utilisée pour l'alias de gestionnaire de files d'attente est le nom du gestionnaire de files d'attente local.

### **OOBRW**

Ouvrez la file d'attente pour parcourir les messages.

La file d'attente est ouverte pour être utilisée avec les appels MQGET suivants avec l'une des options suivantes:

- MGMBRWF
- GMBRWN
- GMBRWC

Ceci est autorisé même si la file d'attente est actuellement ouverte pour OOINPX. Un appel MQOPEN avec l'option OOBRW établit un curseur de navigation et le positionne logiquement avant le premier message de la file d'attente ; voir la zone *GMOPT* décrite dans «[MQGMO \(options d'extraction de message\) sous IBM i](#)», à la page 1127 pour plus d'informations.

Cette option est valide uniquement pour les files d'attente locales, d'alias et modèles ; elle n'est pas valide pour les files d'attente éloignées, les listes de distribution et les objets qui ne sont pas des files d'attente. Elle n'est pas non plus valide si *ODMN* est le nom d'un alias de gestionnaire de files d'attente ; cela est vrai même si la valeur de l'attribut **RemoteQMgrName** dans la définition locale d'une file d'attente éloignée utilisée pour l'alias de gestionnaire de files d'attente est le nom du gestionnaire de files d'attente local.

### **OOOUT**

Ouvrez une file d'attente pour insérer des messages, ou une rubrique ou une chaîne de rubrique pour publier des messages.

La file d'attente est ouverte pour être utilisée avec les appels MQPUT suivants.

Un appel MQOPEN avec cette option peut aboutir même si l'attribut de file d'attente **InhibitPut** est défini sur QAPUTI (bien que les appels MQPUT suivants échouent lorsque l'attribut est défini sur cette valeur).

Cette option est valide pour tous les types de file d'attente, y compris les listes de distribution et les rubriques.

### **OOINQ**

Ouvrez l'objet pour interroger les attributs.

La file d'attente, la liste de noms, la définition de processus ou le gestionnaire de files d'attente est ouvert pour être utilisé avec les appels MQINQ suivants.

Cette option est valide pour tous les types d'objet autres que les listes de distribution. Elle n'est pas valide si *ODMN* est le nom d'un alias de gestionnaire de files d'attente ; ceci est vrai même si la valeur de l'attribut **RemoteQMgrName** dans la définition locale d'une file d'attente éloignée utilisée pour l'alias de gestionnaire de files d'attente est le nom du gestionnaire de files d'attente local.

### **OOSET**

Ouvrez la file d'attente pour définir les attributs.

La file d'attente est ouverte pour être utilisée avec les appels MQSET suivants.

Cette option est valide pour tous les types de file d'attente autres que les listes de distribution. Elle n'est pas valide si *ODMN* est le nom d'une définition locale d'une file d'attente éloignée ; cela est vrai même si la valeur de l'attribut **RemoteQMGrName** dans la définition locale d'une file d'attente éloignée utilisée pour l'attribution d'alias au gestionnaire de files d'attente est le nom du gestionnaire de files d'attente local.

**Options de liaison:** Les options suivantes s'appliquent lorsque l'objet ouvert est une file d'attente de cluster ; ces options contrôlent la liaison du descripteur de file d'attente à une instance de la file d'attente de cluster:

#### **OOBND0**

Descripteur de liaison à la destination lors de l'ouverture de la file d'attente.

Ainsi, le gestionnaire de files d'attente local lie le descripteur de file d'attente à une instance de la file d'attente de destination lors de l'ouverture de la file d'attente. Par conséquent, tous les messages insérés à l'aide de cet identificateur sont envoyés à la même instance de la file d'attente de destination et par la même route.

Cette option est valide uniquement pour les files d'attente et n'affecte que les files d'attente de cluster. Si cette option est spécifiée pour une file d'attente qui n'est pas une file d'attente de cluster, elle est ignorée.

#### **Nom distinctif OBNND**

Ne vous liez pas à une destination spécifique.

Cette opération arrête le gestionnaire de files d'attente local qui lie le descripteur de file d'attente à une instance de la file d'attente de destination. Par conséquent, des appels MQPUT successifs utilisant cet identificateur peuvent entraîner l'envoi des messages à *différentes* instances de la file d'attente de destination ou à la même instance mais par des routes différentes. Elle permet également à l'instance sélectionnée d'être modifiée ultérieurement par le gestionnaire de files d'attente local, par un gestionnaire de files d'attente éloignées ou par un agent MCA, en fonction des conditions du réseau.

**Remarque :** Les applications client et serveur qui doivent échanger une *série* de messages pour terminer une transaction ne doivent pas utiliser OOBNDN (ou OOBNDQ lorsque *DefBind* a la valeur BNDNOT), car les messages successifs de la série peuvent être envoyés à différentes instances de l'application serveur.

Si OOBROW ou l'une des options OOINP\* est spécifiée pour une file d'attente de cluster, le gestionnaire de files d'attente est forcé de sélectionner l'instance locale de la file d'attente de cluster. Par conséquent, la liaison du descripteur de file d'attente est fixe, même si OOBNDN est spécifié.

Si OOINQ est spécifié avec OOBNDN, les appels MQINQ successifs utilisant ce descripteur peuvent interroger différentes instances de la file d'attente de cluster, bien que toutes les instances aient généralement les mêmes valeurs d'attribut.

OOBNDN est valide uniquement pour les files d'attente et n'affecte que les files d'attente de cluster. Si cette option est spécifiée pour une file d'attente qui n'est pas une file d'attente de cluster, elle est ignorée.

#### **OOBNDQ**

Utilisez la liaison par défaut pour la file d'attente.

Ainsi, le gestionnaire de files d'attente local lie le descripteur de file d'attente de la manière définie par l'attribut de file d'attente **DefBind** . La valeur de cet attribut est BNDOPN ou BNDNOT.

OOBNDQ est la valeur par défaut si OOBND0 et OOBNDN ne sont pas indiqués.

OOBNDQ est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette option soit utilisée avec l'une des deux autres options de liaison, mais comme sa valeur est égale à zéro, cette utilisation ne peut pas être détectée.

**Options de contexte:** Les options suivantes contrôlent le traitement du contexte de message:

## OOSAVA

Sauvegarder le contexte lors de l'extraction du message.

Les informations de contexte sont associées à cet identificateur de file d'attente. Ces informations sont définies à partir du contexte de tout message extrait à l'aide de cet identificateur. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

Ces informations contextuelles peuvent être transmises à un message inséré ultérieurement dans une file d'attente à l'aide des appels MQPUT ou MQPUT1 . Voir les options PMPASI et PMPASA décrites dans «MQPMO (options d'insertion de message) sous IBM i», à la page 1230.

Tant qu'un message n'a pas été extrait, le contexte ne peut pas être transmis à un message en cours d'insertion dans une file d'attente.

Les informations de contexte d'un message extrait à l'aide de l'une des options de navigation GMBRW\* ne sont pas sauvegardées (bien que les zones de contexte du paramètre **MSGDSC** soient définies après une navigation).

Cette option est valide uniquement pour les files d'attente locales, d'alias et modèles ; elle n'est pas valide pour les files d'attente éloignées, les listes de distribution et les objets qui ne sont pas des files d'attente. L'une des options OOINP\* doit être indiquée.

## OOPASI

Autoriser la transmission du contexte d'identité.

Cela permet à l'option PMPASI d'être spécifiée dans le paramètre **PMO** lorsqu'un message est placé dans une file d'attente ; cela donne au message les informations contextuelles d'identité d'une file d'attente d'entrée qui a été ouverte avec l'option OOSAVA. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

L'option OOOUT doit être indiquée.

Cette option est valide pour tous les types de file d'attente, y compris les listes de distribution.

## OOPASA

Autoriser la transmission de tous les contextes.

Cela permet de spécifier l'option PMPASA dans le paramètre **PMO** lorsqu'un message est placé dans une file d'attente ; cela donne au message les informations d'identité et de contexte d'origine d'une file d'attente d'entrée qui a été ouverte avec l'option OOSAVA. Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

Cette option implique OOPASI, qui n'a donc pas besoin d'être spécifié. L'option OOOUT doit être indiquée.

Cette option est valide pour tous les types de file d'attente, y compris les listes de distribution.

## OOSSETI

Autoriser la définition du contexte d'identité.

Cela permet de spécifier l'option PMSETI dans le paramètre **PMO** lorsqu'un message est inséré dans une file d'attente ; cela donne au message les informations de contexte d'identité contenues dans le paramètre **MSGDSC** spécifié dans l'appel MQPUT ou MQPUT1 . Pour plus d'informations sur le contexte de message, voir [Contexte de message](#) et [Contrôle des informations de contexte](#).

Cette option implique OOPASI, qui n'a donc pas besoin d'être spécifié. L'option OOOUT doit être indiquée.

Cette option est valide pour tous les types de file d'attente, y compris les listes de distribution.

## OOSSETA

Autoriser la définition de tous les contextes.

Cela permet à l'option PMSETA d'être spécifiée dans le paramètre **PMO** lorsqu'un message est placé dans une file d'attente ; cela donne au message les informations de contexte d'identité et d'origine contenues dans le paramètre **MSGDSC** spécifié dans l'appel MQPUT ou MQPUT1 .

Pour plus d'informations sur le contexte de message, voir [Contexte de message et Contrôle des informations de contexte](#).

Cette option implique les options suivantes, qui ne doivent donc pas être spécifiées:

- OOPASI
- OOPASA
- OOSSETI

L'option OOOOUT doit être indiquée.

Cette option est valide pour tous les types de file d'attente, y compris les listes de distribution.

**Autres options:** Les options suivantes contrôlent la vérification des autorisations et ce qui se passe lorsque le gestionnaire de files d'attente est mis au repos:

#### OOALTU

Valider avec l'ID utilisateur spécifié.

Indique que la zone *ODAU* du paramètre **OBJDSC** contient un identificateur utilisateur qui doit être utilisé pour valider cet appel MQOPEN. L'appel peut aboutir uniquement si ce *ODAU* est autorisé à ouvrir l'objet avec les options d'accès spécifiées, que l'ID utilisateur sous lequel l'application est exécutée soit ou non autorisé à le faire. Cela ne s'applique pas aux options de contexte spécifiées, qui sont toujours vérifiées par rapport à l'identificateur utilisateur sous lequel l'application s'exécute.

Cette option est valide pour tous les types d'objet.

#### OOFIQ

Echec si le gestionnaire de files d'attente est mis au repos.

Cette option force l'échec de l'appel MQOPEN si le gestionnaire de files d'attente est à l'état de mise au repos.

Cette option est valide pour tous les types d'objet.

#### OORLQ

Entrez le nom de la file d'attente locale qui a été ouverte.

Cette option indique que l'option ResolvedQName dans la structure MQOD (si elle est disponible) doit être entrée avec le nom de la file d'attente locale qui a été ouverte. Le nom ResolvedQMGrsera également entré avec le nom du gestionnaire de files d'attente local hébergeant la file d'attente locale.

Option	Alias ( «1», à la page 1397 )	Local et modèle	Eloigné	Cluster non local	Liste de distribution	Topic
OOINPQ	✓	✓	-	-	-	-
OOINPS	✓	✓	-	-	-	-
OOINPX	✓	✓	-	-	-	-
OOWRW	✓	✓	-	-	-	-
OOOUT	✓	✓	✓	✓	✓	✓
OOINQ	✓	✓	«2», à la page 1397	✓	-	-

Tableau 750. Options MQOPEN valides pour chaque type de file d'attente (suite)

Option	Alias ( «1», à la page 1397 )	Local et modèle	Eloigné	Cluster non local	Liste de distribution	Topic
OASET	✓	✓	«2», à la page 1397	-	-	-
OABNDO ( «3», à la page 1397 )	✓	✓	✓	✓	✓	-
OABNDN ( «3», à la page 1397 )	✓	✓	✓	✓	✓	-
OABNDQ ( «3», à la page 1397 )	✓	✓	✓	✓	✓	-
OOSAVA	✓	✓	-	-	-	-
OOPASI	✓	✓	✓	✓	✓	«5», à la page 1397
OOPASA	✓	✓	✓	✓	✓	«5», à la page 1397
OOSSETI	✓	✓	✓	✓	✓	«5», à la page 1397
OOSSETA	✓	✓	✓	✓	✓	«5», à la page 1397
OOALTU	✓	✓	✓	✓	✓	✓
OOFIQ	✓	✓	✓	✓	✓	✓
OORLQ	✓	✓	✓	✓	-	-

**Remarques :**

1. La validité des options pour les alias dépend de la validité de l'option pour la file d'attente dans laquelle l'alias est résolu.
2. Cette option est valide uniquement pour la définition locale d'une file d'attente éloignée.
3. Cette option peut être spécifiée pour n'importe quel type de file d'attente, mais elle est ignorée si la file d'attente n'est pas une file d'attente de cluster.
4. Cet attribut est ignoré pour une rubrique.
5. Ces attributs peuvent être utilisés avec une rubrique, mais ils affectent uniquement le contexte défini pour le message conservé, et non les zones de contexte envoyées à un abonné.

**HOBJ (entier signé à 10 chiffres)-sortie**

Descripteur d'objet.

Ce descripteur représente l'accès qui a été établi à l'objet. Il doit être spécifié lors des appels de mise en file d'attente de messages suivants qui fonctionnent sur l'objet. Elle cesse d'être valide lorsque l'appel MQCLOSE est émis ou lorsque l'unité de traitement qui définit la portée de l'identificateur s'arrête.

La portée de la poignée est limitée à la plus petite unité de traitement parallèle pris en charge par la plateforme sur laquelle l'application s'exécute ; le descripteur n'est pas valide en dehors de l'unité de traitement parallèle à partir de laquelle l'appel MQOPEN a été émis:

- Sous IBM i, la portée du descripteur est le travail qui émet l'appel.

## CMPCOD (entier signé à 10 chiffres)-sortie

Code achèvement.

Il peut s'agir de :

### CCOK

Achèvement réussi.

### CCWARN

Avertissement (achèvement partiel).

### CCFAIL

Echec de l'appel.

## Déclaration RPG

```
C*..1.....2.....3.....4.....5.....6.....7..
C                               CALLP      MQOPEN(HCONN : OBJDSC : OPTS :
C                               HOBJ : CMPCOD : REASON)
```

La définition de prototype pour l'appel est la suivante:

```
D*..1.....2.....3.....4.....5.....6.....7..
DMQOPEN          PR          EXTPROC('MQOPEN')
D* Connection handle
D HCONN          10I 0 VALUE
D* Object descriptor
D OBJDSC          468A
D* Options that control the action of MQOPEN
D OPTS          10I 0 VALUE
D* Object handle
D HOBJ          10I 0
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CMPCOD
D REASON          10I 0
```

IBM i

## MQPUT (insertion de message) sur IBM i

L'appel MQPUT place un message dans une file d'attente, dans une liste de distribution ou dans une rubrique. La file d'attente, la liste de distribution ou la rubrique doit déjà être ouverte.

- [«Syntaxe»](#), à la page 1398
- [«Remarques sur l'utilisation»](#), à la page 1398
  - [«Rubriques»](#), à la page 1399
  - [«MQPUT et MQPUT1»](#), à la page 1399
  - [«Files d'attente de destination»](#), à la page 1399
  - [«Listes de diffusion»](#), à la page 1400
  - [«En-têtes»](#), à la page 1402
  - [«Mémoire tampon»](#), à la page 1403
- [«Paramètres»](#), à la page 1403
- [«Déclaration RPG»](#), à la page 1408

## Syntaxe

MQPUT (HCONN, HOBJ, MSGDSC, PMO, BUFLN, BUFFER, CMPCOD, REASON)

## Remarques sur l'utilisation

## Rubriques

Les remarques suivantes s'appliquent à l'utilisation des rubriques:

1. Lors de l'utilisation de MQPUT pour publier des messages sur une rubrique, où un ou plusieurs abonnés à cette rubrique ne peuvent pas recevoir la publication en raison d'un problème avec leur file d'attente d'abonné (par exemple, elle est saturée), le code anomalie renvoyé à l'appel MQPUT et le comportement de distribution dépendent de la définition des attributs PMSGDLV ou NPMSGDLV sur le TOPIC. Notez que la distribution d'une publication dans la file d'attente des messages non livrés lorsque RODLQ est indiqué, ou la suppression du message lorsque RODISC est indiqué, est considérée comme une distribution réussie du message. Si aucune des publications n'est distribuée, MQPUT renvoie RC2502. Cela peut se produire dans les cas suivants:
  - Un message est publié sur un TOPIC avec PMSGDLV ou NPMSGDLV (en fonction de la persistance du message) défini sur ALL et tout abonnement (durable ou non) possède une file d'attente qui ne peut pas recevoir la publication.
  - Un message est publié sur un TOPIC avec PMSGDLV ou NPMSGDLV (en fonction de la persistance du message) défini sur ALLDUR et un abonnement durable possède une file d'attente qui ne peut pas recevoir la publication.

Le MQPUT peut être renvoyé avec RCNONE même si des publications n'ont pas pu être distribuées à certains abonnés dans les cas suivants:

  - Un message est publié sur un TOPIC avec PMSGDLV ou NPMSGDLV (en fonction de la persistance du message) défini sur ALLAVAIL et tout abonnement, durable ou non, possède une file d'attente qui ne peut pas recevoir la publication.
  - Un message est publié sur un TOPIC avec PMSGDLV ou NPMSGDLV (en fonction de la persistance du message) défini sur ALLDUR et un abonnement non durable possède une file d'attente qui ne peut pas recevoir la publication.
2. S'il n'y a pas d'abonnés à la rubrique utilisée, le message publié n'est envoyé à aucune file d'attente et est supprimé. Il ne fait aucune différence que ce message soit persistant ou non persistant, ou qu'il ait une durée d'expiration illimitée ou une durée d'expiration limitée, il est tout de même supprimé s'il n'y a pas d'abonnés. La seule exception à cette règle est si le message doit être conservé, auquel cas, bien qu'il ne soit pas envoyé aux files d'attente des abonnés, il est stocké par rapport à la rubrique à distribuer aux nouveaux abonnements ou aux abonnés qui demandent des publications conservées à l'aide de MQSUBRQ.

## MQPUT et MQPUT1

Les appels MQPUT et MQPUT1 peuvent être utilisés pour placer des messages dans une file d'attente ; l'appel à utiliser dépend des circonstances

- L'appel MQPUT doit être utilisé lorsque plusieurs messages doivent être placés dans la *même* file d'attente.

Un appel MQOPEN spécifiant l'option OOOUT est émis en premier, suivi d'une ou de plusieurs demandes MQPUT pour ajouter des messages à la file d'attente ; enfin, la file d'attente est fermée avec un appel MQCLOSE. Cela offre de meilleures performances que l'utilisation répétée de l'appel MQPUT1 .

- L'appel MQPUT1 doit être utilisé lorsqu'un seul *message* doit être inséré dans une file d'attente.

Cet appel encapsule les appels MQOPEN, MQPUT et MQCLOSE en un seul appel, ce qui réduit le nombre d'appels qui doivent être émis.

## Files d'attente de destination

Si une application place une séquence de messages dans la même file d'attente sans utiliser de groupes de messages, l'ordre de ces messages est conservé si les conditions suivantes sont remplies. Certaines conditions s'appliquent aux files d'attente de destination locales et éloignées ; d'autres conditions s'appliquent uniquement aux files d'attente de destination éloignées.

## Conditions pour les files d'attente de destination locales et distantes

- Tous les appels MQPUT se trouvent dans la même unité de travail, ou aucun d'entre eux ne se trouve dans une unité de travail.

Lorsque des messages sont placés dans une file d'attente particulière au sein d'une seule unité d'oeuvre, les messages provenant d'autres applications peuvent être entrecoupés avec la séquence de messages de la file d'attente.

- Tous les appels MQPUT sont effectués à l'aide du même descripteur d'objet *HOBJ*.

Dans certains environnements, la séquence de messages est également préservée lorsque des descripteurs d'objet différents sont utilisés, à condition que les appels soient effectués à partir de la même application. La signification de "même application" est déterminée par l'environnement:

– Sous IBM i, l'application est le travail.

- Les messages ont tous la même priorité.

## Conditions supplémentaires pour les files d'attente de destination éloignées

- Il n'existe qu'un seul chemin entre le gestionnaire de files d'attente émetteur et le gestionnaire de files d'attente cible.

S'il est possible que certains messages de la séquence soient placés sur un chemin différent (par exemple, en raison de la reconfiguration, de l'équilibrage du trafic ou de la sélection du chemin en fonction de la taille des messages), l'ordre des messages sur le gestionnaire de files d'attente de destination ne peut pas être garanti.

- Les messages ne sont pas placés temporairement dans des files d'attente de rebut au niveau des gestionnaires de files d'attente d'envoi, intermédiaire ou de destination.

Si un ou plusieurs messages sont placés temporairement dans une file d'attente de rebut (par exemple, parce qu'une file d'attente de transmission ou la file d'attente de destination est temporairement saturée), les messages peuvent arriver dans la file d'attente de destination hors séquence.

- Les messages sont tous persistants ou non persistants.

Si l'attribut **CDNPM** d'un canal sur la route entre les gestionnaires de files d'attente d'envoi et de destination est défini sur *NPFAST*, les messages non persistants peuvent passer avant les messages persistants, ce qui a pour conséquence de ne pas conserver l'ordre des messages persistants par rapport aux messages non persistants. Cependant, l'ordre des messages persistants les uns par rapport aux autres et des messages non persistants les uns par rapport aux autres est conservé.

Si ces conditions ne sont pas satisfaites, les groupes de messages peuvent être utilisés pour préserver l'ordre des messages, mais notez que cela nécessite que les applications d'envoi et de réception utilisent la prise en charge du regroupement de messages. Pour plus d'informations sur les groupes de messages, voir :

- Zone *MDMFL* dans *MQMD*
- Option *PMLOGO* dans *MQPMO*
- Option *GMLOGO* dans *MQGMO*

## Listes de diffusion

Les remarques suivantes s'appliquent à l'utilisation des listes de distribution.

1. Les messages peuvent être placés dans une liste de distribution à l'aide d'une instance version-1 ou d'une instance version-2 *MQPMO*. Si un *MQPMO* version-1 est utilisé (ou un *MQPMO* version-2 avec *PMREC* égal à zéro), aucun enregistrement de message d'insertion ou de réponse ne peut être fourni par l'application. Cela signifie qu'il ne sera pas possible d'identifier les files d'attente qui rencontrent des erreurs, si le message est envoyé avec succès à certaines files d'attente de la liste de distribution et non à d'autres.

Si des enregistrements de message d'insertion ou de réponse sont fournis par l'application, la zone *PMVER* doit être définie sur *PMVER2*.



Un MQPMO version-2 peut également être utilisé pour envoyer des messages à une file d'attente unique qui ne figure pas dans une liste de distribution, en vérifiant que *PMREC* est égal à zéro.

2. Les paramètres de code achèvement et de code anomalie sont définis comme suit:

- Si les insertions dans les files d'attente de la liste de distribution aboutissent ou échouent de la même manière, les paramètres de code achèvement et de code raison sont définis pour décrire le résultat commun. Les enregistrements de réponse MQRR (s'ils sont fournis par l'application) ne sont pas définis dans ce cas.

Par exemple, si chaque insertion aboutit, le code achèvement est défini sur CCOK et le code raison est RCNONE ; si chaque insertion échoue car toutes les files d'attente sont interdites pour les insertions, les paramètres sont définis sur CCFAIL et RC2051.

- Si les insertions dans les files d'attente de la liste de distribution n'aboutissent pas toutes ou échouent de la même manière:
  - Le paramètre de code achèvement est défini sur CCWARN si au moins une insertion a abouti et sur CCFAIL si toutes les opérations ont échoué.
  - Le paramètre de code anomalie est défini sur RC2136.
  - Les enregistrements de réponse (s'ils sont fournis par l'application) sont définis sur les codes achèvement individuels et les codes raison des files d'attente de la liste de distribution.

Si l'insertion dans une destination échoue en raison de l'échec de l'ouverture de cette destination, les zones de l'enregistrement de réponse sont définies sur CCFAIL et RC2137; cette destination est incluse dans *PMIDC*.

3. Si une destination de la liste de distribution se résout en une file d'attente locale, le message est placé dans cette file d'attente sous la forme normale (c'est-à-dire qu'il ne s'agit pas d'un message de liste de distribution). Si plusieurs destinations sont résolues dans la même file d'attente locale, un message est placé dans la file d'attente pour chaque destination.

Si une destination de la liste de distribution est résolue en file d'attente éloignée, un message est placé dans la file d'attente de transmission appropriée. Lorsque plusieurs destinations sont résolues dans la même file d'attente de transmission, un seul message de liste de distribution contenant ces destinations peut être placé dans la file d'attente de transmission, même si ces destinations n'étaient pas adjacentes dans la liste des destinations fournie par l'application. Toutefois, cette opération ne peut être effectuée que si la file d'attente de transmission prend en charge les messages de liste de distribution (voir l'attribut de file d'attente **DistLists** décrit dans [«Attributs des files d'attente»](#), à la page 1435).

Si la file d'attente de transmission ne prend pas en charge les listes de distribution, une copie du message au format normal est placée dans la file d'attente de transmission pour chaque destination qui utilise cette file d'attente de transmission.

Si une liste de distribution avec les données de message d'application est trop grande pour une file d'attente de transmission, le message de liste de distribution est divisé en messages de liste de distribution plus petits, chacun contenant moins de destinations. Si les données de message d'application ne correspondent qu'à la file d'attente, les messages de liste de distribution ne peuvent pas être utilisés du tout et le gestionnaire de files d'attente génère une copie du message au format normal pour chaque destination qui utilise cette file d'attente de transmission.

Si des destinations différentes ont une priorité de message ou une persistance de message différente (cela peut se produire lorsque l'application indique PRQDEF ou PEQDEF), les messages ne sont pas conservés dans le même message de liste de distribution. Au lieu de cela, le gestionnaire de files d'attente génère autant de messages de liste de distribution que nécessaire pour prendre en charge les valeurs de priorité et de persistance différentes.

4. Une insertion dans une liste de distribution peut entraîner:

- un message de liste de distribution unique, ou
- Un certain nombre de messages de liste de distribution plus petits, ou
- Une combinaison de messages de liste de distribution et de messages normaux, ou

- Messages normaux uniquement.

L'occurrence précédente varie selon que:

- Les destinations de la liste sont locales, distantes ou un mélange.
- Les destinations ont la même priorité de message et la même persistance de message.
- Les files d'attente de transmission peuvent contenir des messages de liste de distribution.
- La longueur maximale des messages des files d'attente de transmission est suffisante pour accueillir le message sous forme de liste de distribution.

Toutefois, quel que soit le message ci-dessus, chaque message *physique* résultant (c'est-à-dire chaque message normal ou de liste de distribution résultant de l'insertion) compte comme *un seul* message lorsque:

- Vérifier si l'application a dépassé le nombre maximal de messages autorisé dans une unité de travail (voir l'attribut du gestionnaire de files d'attente **MaxUncommittedMsgs** ).
  - Vérification du respect des conditions de déclenchement.
  - Incrémentation du nombre de lignes de la file d'attente et vérification du dépassement du nombre maximal de lignes de la file d'attente.
5. Toute modification apportée aux définitions de file d'attente qui aurait entraîné la non-validité d'un descripteur si les files d'attente avaient été ouvertes individuellement (par exemple, une modification du chemin de résolution) n'entraîne pas la non-validité du descripteur de liste de distribution. Toutefois, elle entraîne un échec pour cette file d'attente particulière lorsque le descripteur de liste de distribution est utilisé lors d'un appel MQPUT ultérieur.

## En-têtes

Si un message est inséré avec une ou plusieurs structures d'en-tête IBM MQ au début des données de message d'application, le gestionnaire de files d'attente effectue certaines vérifications sur les structures d'en-tête pour vérifier qu'elles sont valides. Si le gestionnaire de files d'attente détecte une erreur, l'appel échoue avec un code anomalie approprié. Les contrôles effectués varient en fonction des structures particulières présentes. En outre, les vérifications sont effectuées uniquement si un MQMD version-2 ou version ultérieure est utilisé sur l'appel MQPUT ou MQPUT1 ; les vérifications ne sont pas effectuées si un MQMD version-1 est utilisé, même si un MQMDE est présent au début des données de message d'application.

Les structures d'en-tête IBM MQ suivantes sont entièrement validées par le gestionnaire de files d'attente: MQDH, MQMDE.

Pour les autres structures d'en-tête IBM MQ , le gestionnaire de files d'attente effectue une validation, mais ne vérifie pas toutes les zones. Les structures qui ne sont pas prises en charge par le gestionnaire de files d'attente local et les structures qui suivent le premier MQDLH dans le message ne sont pas validées.

Outre les vérifications générales des champs dans les structures IBM MQ , les conditions suivantes doivent être remplies:

- Une structure IBM MQ ne doit pas être divisée sur deux ou plusieurs segments-la structure doit être entièrement contenue dans un segment.
- La somme des longueurs des structures d'un message PCF doit être égale à la longueur spécifiée par le paramètre **BUFLEN** dans l'appel MQPUT ou MQPUT1 . Un message PCF est un message comportant l'un des noms de format suivants:
  - FMADMN
  - FMEVNT
  - FMPCF
- Les structures IBM MQ ne doivent pas être tronquées, sauf dans les cas suivants où les structures tronquées sont autorisées:
  - Messages qui sont des messages de rapport.

- Messages PCF.
- Messages contenant une structure MQDLH. (Les structures *suivant* le premier MQDLH peuvent être tronquées ; les structures précédant le MQDLH ne peuvent pas.)

## Mémoire tampon

Le paramètre **BUFFER** indiqué dans l'exemple de programmation RPG est déclaré sous la forme d'une chaîne, ce qui limite la longueur maximale du paramètre à 256 octets. Si une mémoire tampon plus importante est requise, le paramètre doit être déclaré en tant que structure ou zone dans un fichier physique. Cela augmente la longueur maximale possible à environ 32 Ko.

## Paramètres

L'appel MQPUT comporte les paramètres suivants:

### HCONN (entier signé à 10 chiffres)-entrée

Descripteur de connexion.

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *HCONN* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

### HOBJ (entier signé à 10 chiffres)-entrée

Descripteur d'objet.

Cet identificateur représente la file d'attente à laquelle le message est ajouté ou la rubrique dans laquelle le message est publié. La valeur de *HOBJ* a été renvoyée par un appel MQOPEN précédent qui a indiqué l'option OOOOUT.

### MSGDSC (MQMD)-entrée / sortie

Descripteur de message.

Cette structure décrit les attributs du message envoyé et reçoit des informations sur le message une fois la demande d'insertion terminée. Pour plus d'informations, voir «MQMD (descripteur de message) sous IBM i», à la page 1162.

Si l'application fournit un MQMD version-1, les données de message peuvent être préfixées avec une structure MQMDE afin de spécifier des valeurs pour les zones qui existent dans le MQMD version-2 mais pas dans version-1. La zone *MDFMT* du MQMD doit être définie sur FMMDE pour indiquer qu'un MQMDE est présent. Pour plus d'informations, voir «MQMDE (extension de descripteur de message) sous IBM i», à la page 1209.

### PMO (MQPMO)-entrée/sortie

Options qui contrôlent l'action de MQPUT.

Pour plus d'informations, voir «MQPMO (options d'insertion de message) sous IBM i», à la page 1230.

### BUFLEN (entier signé à 10 chiffres)-entrée

Longueur du message dans *BUFFER*.

La valeur zéro est valide et indique que le message ne contient aucune donnée d'application. La limite supérieure de *BUFLEN* dépend de différents facteurs:

- Si la file d'attente de destination est une file d'attente partagée, la limite supérieure est de 63 Ko (64 512 octets).
- Si la destination est une file d'attente locale ou est convertie en file d'attente locale (mais n'est pas une file d'attente partagée), la limite supérieure varie selon que:
  - Le gestionnaire de files d'attente local prend en charge la segmentation.
  - L'application émettrice spécifie l'indicateur qui permet au gestionnaire de files d'attente de segmenter le message. Cet indicateur est MFSEGA et peut être spécifié dans un MQMD version-2 ou dans un MQMDE utilisé avec un MQMD version-1.

Si ces deux conditions sont satisfaites, *BUFLEN* ne peut pas dépasser 999 999 999 moins la valeur de la zone *MDOFF* dans MQMD. Le message logique le plus long pouvant être inséré est donc 999 999 999 octets (lorsque *MDOFF* est égal à zéro). Toutefois, les contraintes de ressources imposées par le système d'exploitation ou l'environnement dans lequel l'application s'exécute peuvent entraîner une limite inférieure.

Si l'une des conditions décrites précédemment ou les deux ne sont pas remplies, *BUFLEN* ne peut pas dépasser la plus petite des valeurs de l'attribut **MaxMsgLength** de la file d'attente et de l'attribut **MaxMsgLength** du gestionnaire de files d'attente.

- Si la destination est une file d'attente éloignée ou qu'elle est résolue en file d'attente éloignée, les conditions des files d'attente locales s'appliquent, *mais au niveau de chaque gestionnaire de files d'attente via lequel le message doit être transmis pour atteindre la file d'attente de destination* ; en particulier:
  1. File d'attente de transmission locale utilisée pour stocker temporairement le message sur le gestionnaire de files d'attente local
  2. Files d'attente de transmission intermédiaires (le cas échéant) utilisées pour stocker les messages au niveau des gestionnaires de files d'attente sur la route entre les gestionnaires de files d'attente local et de destination
  3. File d'attente de destination sur le gestionnaire de files d'attente de destination

Le message le plus long pouvant être inséré est donc régi par les files d'attente et les gestionnaires de files d'attente les plus restrictifs.

Lorsqu'un message se trouve dans une file d'attente de transmission, des informations supplémentaires se trouvent avec les données de message, ce qui réduit la quantité de données d'application pouvant être transférées. Dans ce cas, il est recommandé de soustraire les octets LNMHD des valeurs *MaxMsgLength* des files d'attente de transmission lors de la détermination de la limite pour *BUFLEN*.

**Remarque :** Seul le non-respect de la condition 1 peut être diagnostiqué de manière synchrone (avec le code anomalie RC2030 ou RC2031) lorsque le message est inséré. Si les conditions 2 ou 3 ne sont pas satisfaites, le message est redirigé vers une file d'attente de rebut (message non distribué), soit au niveau d'un gestionnaire de files d'attente intermédiaire, soit au niveau du gestionnaire de files d'attente de destination. Si cela se produit, un message de rapport est généré si l'expéditeur en a demandé un.

### **BUFFER (chaîne de 1 octet x BUFLen)-entrée**

Données de message.

Il s'agit d'une mémoire tampon contenant les données d'application à envoyer. La mémoire tampon doit être alignée sur une limite appropriée à la nature des données du message. L'alignement sur 4 octets doit convenir à la plupart des messages (y compris les messages contenant des structures d'en-tête MQ), mais certains messages peuvent nécessiter un alignement plus strict. Par exemple, un message contenant un entier binaire de 64 bits peut nécessiter un alignement de 8 octets.

Si *BUFFER* contient des données de type caractères, des données numériques ou les deux, les zones *MDCSI* et *MDENC* du paramètre **MSGDSC** doivent être définies sur les valeurs appropriées aux données ; cela permet au destinataire du message de convertir les données (si nécessaire) en jeu de caractères et en codage utilisés par le destinataire.

**Remarque :** Tous les autres paramètres de l'appel MQPUT doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par ENNAT.

### **CMPCOD (entier signé à 10 chiffres)-sortie**

Code achèvement.

Il peut s'agir de :

**CCOK**

Achèvement réussi.

**CCWARN**

Avertissement (achèvement partiel).

**CCFAIL**

Echec de l'appel.

**REASON (entier signé à 10 chiffres)-sortie**

Code anomalie qualifiant *CMPCOD*.

Si *CMPCOD* est CCOK:

**RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CMPCOD* est CCWARN:

**RC2104**

(2104, X'838') L'option de rapport dans le descripteur de message n'est pas reconnue.

**RC2136**

(2136, X'858') Plusieurs codes anomalie ont été renvoyés.

Si *CMPCOD* est CCFAIL:

**RC2004**

(2004, X'7D4') Paramètre de tampon non valide.

**RC2005**

(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

**RC2009**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

**RC2013**

(2013, X'7DD') Heure d'expiration non valide.

**RC2014**

(2014, X'7DE') Code retour non valide.

**RC2018**

(2018, X'7E2') Descripteur de connexion non valide.

**RC2019**

(2019, X'7E3') Descripteur d'objet non valide.

**RC2024**

(2024, X'7E8') Aucun autre message pouvant être traité au sein de l'unité d'oeuvre en cours.

**RC2026**

(2026, X'7EA') Descripteur de message non valide.

**RC2027**

(2027, X'7EB') File d'attente de réponse manquante.

**RC2029**

(2029, X'7ED') Le type de message dans le descripteur de message n'est pas valide.

**RC2030**

(2030, X'7EE') Longueur de message supérieure au maximum pour la file d'attente.

**RC2031**

(2031, X'7EF') Longueur de message supérieure à la longueur maximale pour le gestionnaire de files d'attente.

**RC2039**

(2039, X'7F7') File d'attente non ouverte pour la sortie.

**RC2041**

(2041, X'7F9') Définition d'objet modifiée depuis son ouverture.

**RC2046**

(2046, X'7FE') Options incorrectes ou incohérentes.

- RC2047**  
(2047, X'7FF') Persistance non valide.
- RC2048**  
(2048, X'800') La file d'attente ne prend pas en charge les messages persistants.
- RC2050**  
(2050, X'802') Priorité de message non valide.
- RC2051**  
(2051, X'803') Appels d'insertion bloqués pour la file d'attente.
- RC2052**  
(2052, X'804') File d'attente supprimée.
- RC2053**  
(2053, X'805') La file d'attente contient déjà le nombre maximal de messages.
- RC2056**  
(2056, X'808') Aucun espace disponible sur le disque pour la file d'attente.
- RC2058**  
(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.
- RC2059**  
(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.
- RC2061**  
(2061, X'80D') Les options de rapport dans le descripteur de message ne sont pas valides.
- RC2071**  
(2071, X'817') Mémoire disponible insuffisante.
- RC2072**  
(2072, X'818') Prise en charge du point de synchronisation non disponible.
- RC2093**  
(2093, X'82D') File d'attente non ouverte pour le passage de tous les contextes.
- RC2094**  
(2094, X'82E') File d'attente non ouverte pour le contexte d'identité de transmission.
- RC2095**  
(2095, X'82F') File d'attente non ouverte pour tous les contextes.
- RC2096**  
(2096, X'830') File d'attente non ouverte pour le contexte d'identité défini.
- RC2097**  
(2097, X'831') L'identificateur de file d'attente auquel il est fait référence ne sauvegarde pas le contexte.
- RC2098**  
(2098, X'832') Contexte non disponible pour le descripteur de file d'attente référencé.
- RC2101**  
(2101, X'835') Objet endommagé.
- RC2102**  
(2102, X'836') Ressources système disponibles insuffisantes.
- RC2135**  
(2135, X'857') Structure d'en-tête de distribution incorrecte.
- RC2136**  
(2136, X'858') Plusieurs codes anomalie ont été renvoyés.
- RC2137**  
(2137, X'859') L'ouverture de l'objet a échoué.
- RC2149**  
(2149, X'865') Structures PCF incorrectes.

- RC2154**  
(2154, X'86A') Nombre d'enregistrements présents non valide.
- RC2156**  
(2156, X'86C') Enregistrements de réponse non valides.
- RC2158**  
(2158, X'86E') Les indicateurs d'enregistrement de message d'insertion ne sont pas valides.
- RC2159**  
(2159, X'86F') Enregistrements de message d'insertion non valides.
- RC2161**  
(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.
- RC2162**  
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.
- RC2173**  
(2173, X'87D') La structure des options d'insertion de message n'est pas valide.
- RC2185**  
(2185, X'889') Spécification de persistance incohérente.
- RC2188**  
(2188, X'88C') Appel rejeté par l'exit de charge de travail du cluster.
- RC2189**  
(2189, X'88D') La résolution du nom de cluster a échoué.
- RC2195**  
(2195, X'893') Une erreur inattendue s'est produite.
- RC2219**  
(2219, X'8AB') Appel MQI entré à nouveau avant la fin de l'appel précédent.
- RC2241**  
(2241, X'8C1') Groupe de messages non complet.
- RC2242**  
(2242, X'8C2') Message logique non complet.
- RC2245**  
(2245, X'8C5') Spécification d'unité d'oeuvre incohérente.
- RC2248**  
(2248, X'8C8') Extension de descripteur de message non valide.
- RC2249**  
(2249, X'8C9') Indicateurs de message non valides.
- RC2250**  
(2250, X'8CA') Numéro de séquence de message non valide.
- RC2251**  
(2251, X'8CB') Position de segment de message non valide.
- RC2252**  
(2252, X'8CC') Longueur d'origine incorrecte.
- RC2253**  
(2253, X'8CD') La longueur des données dans le segment de message est égale à zéro.
- RC2255**  
(2255, X'8CF') Unité d'oeuvre non disponible pour le gestionnaire de files d'attente à utiliser.
- RC2257**  
(2257, X'8D1') Version MQMD fournie non valide.
- RC2258**  
(2258, X'8D2') Identificateur de groupe non valide.
- RC2266**  
(2266, X'8DA') L'exit de charge de travail de cluster a échoué.

**RC2269**

(2269, X'8DD') Erreur de ressource de cluster.

**RC2270**

(2270, X'8DE') Aucune file d'attente de destination disponible.

**RC2420**

(2420) Un appel MQPUT a été émis, mais les données de message contiennent une structure MQEPH non valide.

**RC2479**

(2479, X'9AF') La publication n'a pas pu être conservée.

**RC2480**

(2480, X'9B0') Le type de cible a été modifié: la file d'attente alias fait référence à une file d'attente mais fait désormais référence à une rubrique.

**RC2502**

(2502, X'9C6') La publication a échoué et la publication n'a été distribuée à aucun abonné

**RC2551**

(2551, X'9F7') La chaîne de sélection spécifiée n'est pas disponible.

**RC2554**

(2554, X'9FA') Le contenu du message n'a pas pu être analysé pour déterminer si le message doit être distribué à un abonné avec un sélecteur de message étendu.

**Déclaration RPG**

```

C*.1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQPUT(HCONN : HOBJ : MSGDSC : PMO :
C                      BUFLen : BUFFER : CMPCOD :
C                      REASON)

```

La définition de prototype pour l'appel est la suivante:

```

D*.1.....2.....3.....4.....5.....6.....7..
DMQPUT      PR          EXTPROC('MQPUT')
D* Connection handle
D HCONN          10I 0 VALUE
D* Object handle
D HOBJ          10I 0 VALUE
D* Message descriptor
D MSGDSC          364A
D* Options that control the action of MQPUT
D PMO          200A
D* Length of the message in Buffer
D BUFLen          10I 0 VALUE
D* Message data
D BUFFER          * VALUE
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CMPCOD
D REASON          10I 0

```

**IBM i MQPUT1 (insertion d'un message) sous IBM i**

L'appel MQPUT1 insère un message dans une file d'attente ou une liste de distribution ou dans une rubrique. Il n'est pas nécessaire d'ouvrir la file d'attente, la liste de distribution ou la rubrique.

- «Syntaxe», à la page 1409
- «Remarques sur l'utilisation», à la page 1409
- «Paramètres», à la page 1410
- «Déclaration RPG», à la page 1414



## Syntaxe

MQPUT1 (HCONN, OBJDSC, MSGDSC, PMO, BUFLen, BUFFER, CMPCOD, REASON)

## Remarques sur l'utilisation

1. Les appels MQPUT et MQPUT1 peuvent être utilisés pour placer des messages dans une file d'attente ; l'appel à utiliser dépend des circonstances:

- L'appel MQPUT doit être utilisé lorsque plusieurs messages doivent être placés dans la *même* file d'attente.

Un appel MQOPEN spécifiant l'option OOOUT est émis en premier, suivi d'une ou de plusieurs demandes MQPUT pour ajouter des messages à la file d'attente ; enfin, la file d'attente est fermée avec un appel MQCLOSE. Cela offre de meilleures performances que l'utilisation répétée de l'appel MQPUT1 .

- L'appel MQPUT1 doit être utilisé lorsqu'un seul *message* doit être inséré dans une file d'attente.

Cet appel encapsule les appels MQOPEN, MQPUT et MQCLOSE en un seul appel, ce qui réduit le nombre d'appels qui doivent être émis.

2. Si une application place une séquence de messages dans la même file d'attente sans utiliser de groupes de messages, l'ordre de ces messages est préservé sous réserve que certaines conditions soient satisfaites. Toutefois, dans la plupart des environnements, l'appel MQPUT1 ne répond pas à ces conditions et ne préserve donc pas l'ordre des messages. L'appel MQPUT doit être utilisé à la place dans ces environnements. Pour plus de détails, voir les remarques sur l'utilisation dans la description de l'appel MQPUT.
3. L'appel MQPUT1 peut être utilisé pour insérer des messages dans des listes de distribution. Pour des informations générales à ce sujet, voir les remarques sur la syntaxe des appels MQOPEN et MQPUT.

Les différences suivantes s'appliquent lors de l'utilisation de l'appel MQPUT1 :

- a. Si des enregistrements de réponse MQRR sont fournis par l'application, ils doivent être fournis à l'aide de la structure MQOD ; ils ne peuvent pas être fournis à l'aide de la structure MQPMO.
- b. Le code anomalie RC2137 n'est jamais renvoyé par MQPUT1 dans les enregistrements de réponse ; si l'ouverture d'une file d'attente échoue, l'enregistrement de réponse de cette file d'attente contient le code anomalie réel résultant de l'opération d'ouverture.

Si une opération d'ouverture d'une file d'attente aboutit avec le code achèvement CCWARN, le code achèvement et le code anomalie de l'enregistrement de réponse de cette file d'attente sont remplacés par les codes achèvement et anomalie résultant de l'opération d'insertion.

Comme pour les appels MQOPEN et MQPUT, le gestionnaire de files d'attente définit les enregistrements de réponse (s'ils sont fournis) uniquement lorsque le résultat de l'appel n'est pas le même pour toutes les files d'attente de la liste de distribution ; cela est indiqué par l'exécution de l'appel avec le code anomalie RC2136.

4. Si l'appel MQPUT1 est utilisé pour placer un message dans une file d'attente de cluster, l'appel se comporte comme si OOBNDN avait été spécifié dans l'appel MQOPEN.
5. Si un message est inséré avec une ou plusieurs structures d'en-tête IBM MQ au début des données de message d'application, le gestionnaire de files d'attente effectue certaines vérifications sur les structures d'en-tête pour vérifier qu'elles sont valides. Pour plus d'informations à ce sujet, voir les remarques sur l'utilisation de l'appel MQPUT.
6. Si plusieurs situations d'avertissement se produisent (voir le paramètre **CMPCOD**), le code anomalie renvoyé est le *premier* de la liste suivante qui s'applique:
  - a. RC2136
  - b. RC2242
  - c. RC2241
  - d. RC2049 ou RC2104

7. Le paramètre **BUFFER** indiqué dans l'exemple de programmation RPG est déclaré sous la forme d'une chaîne, ce qui limite la longueur maximale du paramètre à 256 octets. Si une mémoire tampon plus importante est requise, le paramètre doit être déclaré en tant que structure ou zone dans un fichier physique. Cela augmente la longueur maximale possible à environ 32 Ko.

## Paramètres

L'appel MQPUT1 comporte les paramètres suivants:

### **HCONN (entier signé à 10 chiffres)-entrée**

Descripteur de connexion.

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *HCONN* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

### **OBJDSC (MQOD)-entrées-sorties**

Descripteur d'objet.

Il s'agit d'une structure qui identifie la file d'attente à laquelle le message est ajouté. Pour plus d'informations, voir [«MQOD \(descripteur d'objet\) sous IBM i»](#), à la page 1215.

L'utilisateur doit être autorisé à ouvrir la file d'attente pour la sortie. La file d'attente ne doit **pas** être une file d'attente modèle.

### **MSGDSC (MQMD)-entrée / sortie**

Descripteur de message.

Cette structure décrit les attributs du message envoyé et reçoit des informations en retour une fois la demande d'insertion terminée. Pour plus d'informations, voir [«MQMD \(descripteur de message\) sous IBM i»](#), à la page 1162.

Si l'application fournit un MQMD version-1, les données de message peuvent être préfixées avec une structure MQMDE afin de spécifier des valeurs pour les zones qui existent dans le MQMD version-2 mais pas dans version-1. La zone *MDFMT* du MQMD doit être définie sur FMMDE pour indiquer qu'un MQMDE est présent. Pour plus d'informations, voir [«MQMDE \(extension de descripteur de message\) sous IBM i»](#), à la page 1209.

### **PMO (MQPMO)-entrée/sortie**

Options contrôlant l'action de MQPUT1.

Pour plus d'informations, voir [«MQPMO \(options d'insertion de message\) sous IBM i»](#), à la page 1230.

### **BUFLEN (entier signé à 10 chiffres)-entrée**

Longueur du message dans *BUFFER*.

La valeur zéro est valide et indique que le message ne contient aucune donnée d'application. La limite supérieure dépend de différents facteurs ; pour plus de détails, voir la description du paramètre **BUFLEN** de l'appel MQPUT.

### **BUFFER (chaîne de 1 octet x BUFLEN)-entrée**

Données de message.

Il s'agit d'une mémoire tampon contenant les données de message d'application à envoyer. La mémoire tampon doit être alignée sur une limite appropriée à la nature des données du message. L'alignement à 4 octets doit convenir à la plupart des messages (y compris les messages contenant des structures d'en-tête IBM MQ), mais certains messages peuvent nécessiter un alignement plus strict. Par exemple, un message contenant un entier binaire de 64 bits peut nécessiter un alignement de 8 octets.

Si *BUFFER* contient des données de type caractères, des données numériques ou les deux, les zones *MDCSI* et *MDENC* du paramètre **MSGDSC** doivent être définies sur les valeurs appropriées aux

données ; cela permet au destinataire du message de convertir les données (si nécessaire) en jeu de caractères et en codage utilisés par le destinataire.

**Remarque :** Tous les autres paramètres de l'appel MQPUT1 doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et dans le codage du gestionnaire de files d'attente local indiqué par ENNAT.

### **CMPCOD (entier signé à 10 chiffres)-sortie**

Code achèvement.

Il peut s'agir de :

#### **CCOK**

Achèvement réussi.

#### **CCWARN**

Avertissement (achèvement partiel).

#### **CCFAIL**

Echec de l'appel.

### **REASON (entier signé à 10 chiffres)-sortie**

Code anomalie qualifiant *CMPCOD*.

Si *CMPCOD* est CCOK:

#### **RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CMPCOD* est CCWARN:

#### **RC2104**

(2104, X'838') L'option de rapport dans le descripteur de message n'est pas reconnue.

#### **RC2136**

(2136, X'858') Plusieurs codes anomalie ont été renvoyés.

#### **RC2049**

(2049, X'801') La priorité de message dépasse la valeur maximale prise en charge.

#### **RC2241**

(2241, X'8C1') Groupe de messages non complet.

#### **RC2242**

(2242, X'8C2') Message logique non complet.

Si *CMPCOD* est CCFAIL:

#### **RC2001**

(2001, X'7D1') Le type de la file d'attente de base de l'alias n'est pas valide.

#### **RC2004**

(2004, X'7D4') Paramètre de tampon non valide.

#### **RC2005**

(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

#### **RC2009**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

#### **RC2013**

(2013, X'7DD') Heure d'expiration non valide.

#### **RC2014**

(2014, X'7DE') Code retour non valide.

#### **RC2017**

(2017, X'7E1') Plus de descripteurs disponibles.

#### **RC2018**

(2018, X'7E2') Descripteur de connexion non valide.

**RC2024**

(2024, X'7E8') Aucun autre message pouvant être traité au sein de l'unité d'oeuvre en cours.

**RC2026**

(2026, X'7EA') Descripteur de message non valide.

**RC2027**

(2027, X'7EB') File d'attente de réponse manquante.

**RC2029**

(2029, X'7ED') Le type de message dans le descripteur de message n'est pas valide.

**RC2030**

(2030, X'7EE') Longueur de message supérieure au maximum pour la file d'attente.

**RC2031**

(2031, X'7EF') Longueur de message supérieure à la longueur maximale pour le gestionnaire de files d'attente.

**RC2035**

(2035, X'7F3') Non autorisé pour l'accès.

**RC2042**

(2042, X'7FA') Objet déjà ouvert avec des options en conflit.

**RC2043**

(2043, X'7FB') Type d'objet non valide.

**RC2044**

(2044, X'7FC') Structure de descripteur d'objet non valide.

**RC2046**

(2046, X'7FE') Options incorrectes ou incohérentes.

**RC2047**

(2047, X'7FF') Persistance non valide.

**RC2048**

(2048, X'800') La file d'attente ne prend pas en charge les messages persistants.

**RC2050**

(2050, X'802') Priorité de message non valide.

**RC2051**

(2051, X'803') Appels d'insertion bloqués pour la file d'attente.

**RC2052**

(2052, X'804') File d'attente supprimée.

**RC2053**

(2053, X'805') La file d'attente contient déjà le nombre maximal de messages.

**RC2056**

(2056, X'808') Aucun espace disponible sur le disque pour la file d'attente.

**RC2057**

(2057, X'809') Type de file d'attente non valide.

**RC2058**

(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

**RC2059**

(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

**RC2061**

(2061, X'80D') Les options de rapport dans le descripteur de message ne sont pas valides.

**RC2063**

(2063, X'80F') Une erreur de sécurité s'est produite.

**RC2071**

(2071, X'817') Mémoire disponible insuffisante.

- RC2072**  
(2072, X'818') Prise en charge du point de synchronisation non disponible.
- RC2082**  
(2082, X'822') File d'attente de base alias inconnue.
- RC2085**  
(2085, X'825') Nom d'objet inconnu.
- RC2086**  
(2086, X'826') Gestionnaire de files d'attente d'objets inconnu.
- RC2087**  
(2087, X'827') Gestionnaire de files d'attente éloignées inconnu.
- RC2091**  
(2091, X'82B') File d'attente de transmission non locale.
- RC2092**  
(2092, X'82C') File d'attente de transmission avec une utilisation incorrecte.
- RC2097**  
(2097, X'831') L'identificateur de file d'attente auquel il est fait référence ne sauvegarde pas le contexte.
- RC2098**  
(2098, X'832') Contexte non disponible pour le descripteur de file d'attente référencé.
- RC2101**  
(2101, X'835') Objet endommagé.
- RC2102**  
(2102, X'836') Ressources système disponibles insuffisantes.
- RC2135**  
(2135, X'857') Structure d'en-tête de distribution incorrecte.
- RC2136**  
(2136, X'858') Plusieurs codes anomalie ont été renvoyés.
- RC2149**  
(2149, X'865') Structures PCF incorrectes.
- RC2154**  
(2154, X'86A') Nombre d'enregistrements présents non valide.
- RC2155**  
(2155, X'86B') Enregistrements d'objet non valides.
- RC2156**  
(2156, X'86C') Enregistrements de réponse non valides.
- RC2158**  
(2158, X'86E') Les indicateurs d'enregistrement de message d'insertion ne sont pas valides.
- RC2159**  
(2159, X'86F') Enregistrements de message d'insertion non valides.
- RC2161**  
(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.
- RC2162**  
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.
- RC2173**  
(2173, X'87D') La structure des options d'insertion de message n'est pas valide.
- RC2184**  
(2184, X'888') Nom de file d'attente éloignée incorrect.
- RC2188**  
(2188, X'88C') Appel rejeté par l'exit de charge de travail du cluster.

**RC2189**

(2189, X'88D') La résolution du nom de cluster a échoué.

**RC2195**

(2195, X'893') Une erreur inattendue s'est produite.

**RC2196**

(2196, X'894') File d'attente de transmission inconnue.

**RC2197**

(2197, X'895') File d'attente de transmission par défaut inconnue.

**RC2198**

(2198, X'896') La file d'attente de transmission par défaut n'est pas locale.

**RC2199**

(2199, X'897') Erreur d'utilisation de la file d'attente de transmission par défaut.

**RC2258**

(2258, X'8D2') Identificateur de groupe non valide.

**RC2248**

(2248, X'8C8') Extension de descripteur de message non valide.

**RC2219**

(2219, X'8AB') Appel MQI entré à nouveau avant la fin de l'appel précédent.

**RC2249**

(2249, X'8C9') Indicateurs de message non valides.

**RC2250**

(2250, X'8CA') Numéro de séquence de message non valide.

**RC2251**

(2251, X'8CB') Position de segment de message non valide.

**RC2252**

(2252, X'8CC') Longueur d'origine incorrecte.

**RC2253**

(2253, X'8CD') La longueur des données dans le segment de message est égale à zéro.

**RC2255**

(2255, X'8CF') Unité d'oeuvre non disponible pour le gestionnaire de files d'attente à utiliser.

**RC2257**

(2257, X'8D1') Version MQMD fournie non valide.

**RC2266**

(2266, X'8DA') L'exit de charge de travail de cluster a échoué.

**RC2269**

(2269, X'8DD') Erreur de ressource de cluster.

**RC2270**

(2270, X'8DE') Aucune file d'attente de destination disponible.

**RC2420**

(2420) Un appel MQPUT1 a été émis, mais les données de message contiennent une structure MQEPH non valide.

**RC2551**

(2551, X'9F7') La chaîne de sélection spécifiée n'est pas disponible.

**RC2554**

(2554, X'9FA') Le contenu du message n'a pas pu être analysé pour déterminer si le message doit être distribué à un abonné avec un sélecteur de message étendu.

**Déclaration RPG**

C\*..1.....2.....3.....4.....5.....6.....7..

```

C          CALLP      MQPUT1(HCONN : OBJDSC : MSGDSC :
C          PMO : BUFLN : BUFFER :
C          CMPCOD : REASON)

```

La définition de prototype pour l'appel est la suivante:

```

D*..1.....2.....3.....4.....5.....6.....7..
DMQPUT1          PR          EXTPROC('MQPUT1')
D* Connection handle
D HCONN          10I 0 VALUE
D* Object descriptor
D OBJDSC          468A
D* Message descriptor
D MSGDSC          364A
D* Options that control the action of MQPUT1
D PMO            200A
D* Length of the message in BUFFER
D BUFLN          10I 0 VALUE
D* Message data
D BUFFER          * VALUE
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CMPCOD
D REASON          10I 0

```

## IBM i MQSET (définition des attributs d'objet) sous IBM i

L'appel MQSET permet de modifier les attributs d'un objet représenté par un descripteur. L'objet doit être une file d'attente.

- «Syntaxe», à la page 1415
- «Remarques sur l'utilisation», à la page 1415
- «Paramètres», à la page 1416
- «Déclaration RPG», à la page 1419

### Syntaxe

MQSET (*HCONN*, *HOBJ*, *SELCNT*, *SELS*, *IACNT*, *INTATR*, *CALEN*, *CHRATR*, *CMPCOD*, *REASON*)

### Remarques sur l'utilisation

1. A l'aide de cet appel, l'application peut spécifier un tableau d'attributs entiers ou une collection de chaînes d'attributs de caractères, ou les deux. Si aucune erreur ne se produit, les attributs spécifiés sont tous définis simultanément. Si une erreur se produit (par exemple, si un sélecteur n'est pas valide ou qu'une tentative est effectuée pour définir un attribut sur une valeur non valide), l'appel échoue et aucun attribut n'est défini.
2. Les valeurs des attributs peuvent être déterminées à l'aide de l'appel MQINQ ; Pour plus d'informations, voir «MQINQ (interrogation des attributs d'objet) sur IBM i», à la page 1369 .

**Remarque :** Tous les attributs dont les valeurs peuvent être renseignées à l'aide de l'appel MQINQ ne peuvent pas avoir leurs valeurs modifiées à l'aide de l'appel MQSET. Par exemple, aucun attribut d'objet de processus ou de gestionnaire de files d'attente ne peut être défini avec cet appel.

3. Les modifications d'attribut sont conservées lors des redémarrages du gestionnaire de files d'attente (autres que les modifications apportées aux files d'attente dynamiques temporaires qui ne survivent pas aux redémarrages du gestionnaire de files d'attente).
4. Vous ne pouvez pas modifier les attributs d'une file d'attente modèle à l'aide de l'appel MQSET. Toutefois, si vous ouvrez une file d'attente modèle à l'aide de l'appel MQOPEN avec l'option MQOO\_SET, vous pouvez utiliser l'appel MQSET pour définir les attributs de la file d'attente locale dynamique créée par l'appel MQOPEN.

5. Si l'objet défini est une file d'attente de cluster, il doit y avoir une instance locale de la file d'attente de cluster pour que l'ouverture aboutisse.

Pour plus d'informations sur les attributs d'objet, voir:

- «Attributs des files d'attente», à la page 1435
- «Attributs des listes de noms», à la page 1466
- «Attributs des définitions de processus sous IBM i», à la page 1467
- «Attributs du gestionnaire de files d'attente sous IBM i», à la page 1469

## Paramètres

L'appel MQSET comporte les paramètres suivants:

### **HCONN (entier signé à 10 chiffres)-entrée**

Descripteur de connexion.

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de HCONN a été renvoyée par un appel MQCONN ou MQCONNX précédent.

### **HOBJ (entier signé à 10 chiffres)-entrée**

Descripteur d'objet.

Cet identificateur représente l'objet de file d'attente avec les attributs à définir. Le descripteur a été renvoyé par un appel MQOPEN précédent qui a spécifié l'option OOSSET.

### **SELCNT (entier signé à 10 chiffres)-entrée**

Nombre de sélecteurs.

Nombre de sélecteurs fournis dans le tableau SELS . Il s'agit du nombre d'attributs à définir. Zéro est une valeur valide. Le nombre maximal autorisé est 256.

### **SELS (entier signé à 10 chiffres x SELCNT)-entrée**

Tableau de sélecteurs d'attribut.

Il s'agit d'un tableau de sélecteurs d'attribut **SELCNT** ; chaque sélecteur identifie un attribut (entier ou caractère) avec une valeur à définir.

Chaque sélecteur doit être valide pour le type de file d'attente représenté par HOBJ . Seules certaines valeurs IA\* et CA\* sont autorisées ; ces valeurs sont répertoriées plus loin dans cette section.

Les sélecteurs peuvent être spécifiés dans n'importe quel ordre. Les valeurs d'attribut qui correspondent à des sélecteurs d'attribut d'entier (IA\* selectors) doivent être spécifiées dans INTATR dans l'ordre dans lequel ces sélecteurs apparaissent dans SELS. Les valeurs d'attribut qui correspondent à des sélecteurs d'attribut de caractères (sélecteurs CA\*) doivent être spécifiées dans CHRATR dans l'ordre dans lequel ces sélecteurs apparaissent. Les sélecteurs A\* peuvent être entrelacés avec les sélecteurs CA\* ; seul l'ordre relatif au sein de chaque type est important.

Il n'est pas erroné de spécifier le même sélecteur plusieurs fois ; si cela est fait, la dernière valeur spécifiée pour un sélecteur particulier est celle qui prend effet.

#### **Remarque :**

1. Les sélecteurs d'attribut d'entier et de caractère sont alloués dans deux plages différentes ; les sélecteurs IA\* résident dans la plage IAFRST à IALAST, et les sélecteurs CA\* dans la plage CAFRST à CALAST.

Pour chaque plage, les constantes IALSTU et CALSTU définissent la valeur la plus élevée acceptée par le gestionnaire de files d'attente.

2. Si tous les sélecteurs IA\* apparaissent en premier, les mêmes numéros d'éléments peuvent être utilisés pour adresser les éléments correspondants dans les tableaux SELS et INTATR .



Les attributs qui peuvent être définis sont répertoriés dans le tableau suivant. Aucun autre attribut ne peut être défini à l'aide de cet appel. Pour les sélecteurs d'attribut CA\*, la constante qui définit la longueur en octets de la chaîne requise dans CHRATR est fournie entre parenthèses.

<i>Tableau 751. Sélecteurs d'attributs MQSET pour les files d'attente</i>		
<b>Sélecteur</b>	<b>Description</b>	<b>Remarque</b>
CATRGD	Données de déclenchement (LNTRGD).	<u>«2», à la page 1417</u>
IADIST	Prise en charge de la liste de distribution.	<u>«1», à la page 1417</u>
IAIGET	Indique si les opérations d'extraction sont autorisées.	
IAIPUT	Indique si les opérations d'insertion sont autorisées.	
IATRGC	Contrôle du déclencheur.	<u>«2», à la page 1417</u>
IATRGD	Longueur de déclenchement.	<u>«2», à la page 1417</u>
IATRGP	Priorité de message de seuil pour les déclencheurs.	<u>«2», à la page 1417</u>
IATRGT	Type de déclencheur.	<u>«2», à la page 1417</u>

**Remarques :**

1. Pris en charge uniquement sur les plateformes suivantes:

-  AIX
-  IBM i
-  Windows

et pour les clients IBM MQ connectés à ces systèmes.

2. Non pris en charge sur VSE/ESA.

**IACNT (entier signé à 10 chiffres)-entrée**

Nombre d'attributs de type entier.

Il s'agit du nombre d'éléments du tableau INTATR et doit être au moins égal au nombre de sélecteurs IA\* dans le paramètre **SELS** . Zéro est une valeur valide s'il n'y en a pas.

### **INTATR (entier signé à 10 chiffres x rxIACNT)-entrée**

Tableau d'attributs de type entier.

Il s'agit d'un tableau de valeurs d'attribut d'entier IACNT . Ces valeurs d'attribut doivent être dans le même ordre que les sélecteurs IA\* dans le tableau SELS .

### **CALEN (entier signé à 10 chiffres)-entrée**

Longueur de la mémoire tampon des attributs de caractères.

Il s'agit de la longueur en octets du paramètre **CHRATR** , qui doit être au moins la somme des longueurs des attributs de caractères spécifiés dans le tableau SELS . Zéro est une valeur valide s'il n'y a pas de sélecteurs CA\* dans SELS.

### **CHRATR (chaîne de caractères de 1 octet x CALEN)-entrée**

Attributs de caractères.

Il s'agit de la mémoire tampon contenant les valeurs d'attribut de caractère, concaténées. La longueur de la mémoire tampon est indiquée par le paramètre **CALEN** .

Les attributs de caractères doivent être spécifiés dans le même ordre que les sélecteurs CA\* dans le tableau SELS . La longueur de chaque attribut de caractère est fixe (voir SELS). Si la valeur à définir pour un attribut contient moins de caractères non blancs que la longueur définie de l'attribut, la valeur dans CHRATR doit être remplie à droite avec des blancs pour que la valeur de l'attribut corresponde à la longueur définie de l'attribut.

### **CMPCOD (entier signé à 10 chiffres)-sortie**

Code achèvement.

Il peut s'agir de :

#### **CCOK**

Achèvement réussi.

#### **CCFAIL**

Echec de l'appel.

### **REASON (entier signé à 10 chiffres)-sortie**

Code anomalie qualifiant CMPCOD.

Si CMPCOD est CCOK:

#### **RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si CMPCOD est CCFAIL:

#### **RC2219**

(2219, X'8AB') Appel MQI entré à nouveau avant la fin de l'appel précédent.

#### **RC2006**

(2006, X'7D6') Longueur des attributs de caractère non valide.

#### **RC2007**

(2007, X'7D7') Chaîne d'attributs de caractères non valide.

#### **RC2009**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

#### **RC2018**

(2018, X'7E2') Descripteur de connexion non valide.

#### **RC2019**

(2019, X'7E3') Descripteur d'objet non valide.

#### **RC2020**

(2020, X'7E4') Valeur de l'attribut de file d'attente d'interdiction d'extraction ou d'interdiction d'insertion non valide.

**RC2021**

(2021, X'7E5') Nombre d'attributs de type entier non valide.

**RC2023**

(2023, X'7E7') Tableau d'attributs d'entier non valide.

**RC2040**

(2040, X'7F8') File d'attente non ouverte pour l'ensemble.

**RC2041**

(2041, X'7F9') Définition d'objet modifiée depuis son ouverture.

**RC2101**

(2101, X'835') Objet endommagé.

**RC2052**

(2052, X'804') File d'attente supprimée.

**RC2058**

(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

**RC2059**

(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

**RC2162**

(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

**RC2102**

(2102, X'836') Ressources système disponibles insuffisantes.

**RC2065**

(2065, X'811') Nombre de sélecteurs non valide.

**RC2067**

(2067, X'813') Sélecteur d'attribut non valide.

**RC2066**

(2066, X'812') Nombre de sélecteurs trop grand.

**RC2071**

(2071, X'817') Mémoire disponible insuffisante.

**RC2075**

(2075, X'81B') Valeur de l'attribut trigger-control non valide.

**RC2076**

(2076, X'81C') Valeur de l'attribut trigger-depth non valide.

**RC2077**

(2077, X'81D') Valeur de l'attribut trigger-message-priority non valide.

**RC2078**

(2078, X'81E') Valeur de l'attribut de type de déclencheur non valide.

**RC2195**

(2195, X'893') Une erreur inattendue s'est produite.

**Déclaration RPG**

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQSET(HCONN : HOBJ : SELCNT :
C                      SELS(1) : IACNT : INTATR(1) :
C                      CALEN : CHRATR : CMPCOD :
C                      REASON)

```

La définition de prototype pour l'appel est la suivante:

```

D*..1.....2.....3.....4.....5.....6.....7..
DMQSET      PR          EXTPROC('MQSET')
D* Connection handle

```

D HCONN	10I 0 VALUE
D* Object handle	
D HOBJ	10I 0 VALUE
D* Count of selectors	
D SELCNT	10I 0 VALUE
D* Array of attribute selectors	
D SELS	10I 0
D* Count of integer attributes	
D IACNT	10I 0 VALUE
D* Array of integer attributes	
D INTATR	10I 0
D* Length of character attributes buffer	
D CALEN	10I 0 VALUE
D* Character attributes	
D CHRATR	* VALUE
D* Completion code	
D CMPCOD	10I 0
D* Reason code qualifying CMPCOD	
D REASON	10I 0

## MQSETMP (définition de la propriété de descripteur de message) sous IBM i

L'appel MQSETMP définit ou modifie une propriété d'un descripteur de message.

- «Syntaxe», à la page [1420](#)
- «Remarques sur l'utilisation», à la page [1420](#)
- «Paramètres», à la page [1422](#)
- «Déclaration RPG», à la page [1424](#)

### Syntaxe

MQSETMP (*Hconn*, *Hmsg*, *SetPropOpts*, *Name*, *PropDesc*, *Type*, *ValueLength*, *Value*, *CompCode*, *Reason*)

### Remarques sur l'utilisation

- Vous pouvez utiliser cet appel uniquement lorsque le gestionnaire de files d'attente lui-même coordonne l'unité d'oeuvre. Ce nom peut être :
  - Unité de travail locale, dans laquelle les modifications affectent uniquement les ressources IBM MQ .
  - Unité de travail globale dans laquelle les modifications peuvent affecter les ressources appartenant à d'autres gestionnaires de ressources, ainsi que les ressources IBM MQ .

Pour plus de détails sur les unités d'oeuvre locales et globales, voir [«MQBEGIN \(Début de l'unité de travail\) sur IBM i»](#), à la page [1315](#).

- Dans les environnements où le gestionnaire de files d'attente ne coordonne pas l'unité d'oeuvre, utilisez l'appel d'annulation approprié à la place de MQBACK. L'environnement peut également prendre en charge une interruption implicite provoquée par l'arrêt anormal de l'application.
  - Sous z/OS, utilisez les appels suivants:
    - Les programmes batch (y compris les programmes IMS batch DL/I) peuvent utiliser l'appel MQBACK si l'unité de travail n'affecte que les ressources IBM MQ . Toutefois, si l'unité de travail affecte à la fois les ressources IBM MQ et les ressources appartenant à d'autres gestionnaires de ressources (par exemple, Db2 ), utilisez l'appel SRRBACK fourni par le service RRS (z/OS Recoverable Resource Service). L'appel SRRBACK annule les modifications apportées aux ressources appartenant aux gestionnaires de ressources qui ont été activés pour la coordination RRS.
    - Les applications CICS doivent utiliser la commande EXEC CICS SYNCPOINT ROLLBACK pour rétablir l'unité d'oeuvre. N'utilisez pas l'appel MQBACK pour les applications CICS .

- Les applications IMS (autres que les programmes batch DL/I) doivent utiliser des appels IMS tels que ROLB pour rétablir l'unité d'oeuvre. N'utilisez pas l'appel MQBACK pour les applications IMS (autres que les programmes batch DL/I).
- Sous IBM i, utilisez cet appel pour les unités de travail locales coordonnées par le gestionnaire de files d'attente. Cela signifie qu'une définition de validation ne doit pas exister au niveau du travail, c'est-à-dire que la commande STRCMTCTL avec le paramètre **CMTSCOPE (\*JOB)** ne doit pas avoir été émise pour le travail.
- Si une application se termine avec des modifications non validées dans une unité d'oeuvre, la disposition de ces modifications varie selon que l'application se termine normalement ou anormalement. Pour plus de détails, voir les remarques sur l'utilisation dans «MQDISC (gestionnaire de files d'attente de déconnexion) sous IBM i», à la page 1353 .
- Lorsqu'une application insère ou extrait des messages dans des groupes ou des segments de messages logiques, le gestionnaire de files d'attente conserve les informations relatives au groupe de messages et au message logique pour les derniers appels MQPUT et MQGET ayant abouti. Ces informations sont associées à l'identificateur de file d'attente et incluent les éléments suivants:
  - Valeurs des zones *GroupId*, *MsgSeqNumber*, *Offset* et *MsgFlags* dans MQMD.
  - Indique si le message fait partie d'une unité de travail.
  - Pour l'appel MQPUT: indique si le message est persistant ou non persistant.

Le gestionnaire de files d'attente conserve trois ensembles d'informations de groupe et de segment, un ensemble pour chacun des éléments suivants:

- Dernier appel MQPUT réussi (qui peut faire partie d'une unité de travail).
- Dernier appel MQGET réussi qui a supprimé un message de la file d'attente (il peut faire partie d'une unité d'oeuvre).
- Dernier appel MQGET ayant réussi à parcourir un message dans la file d'attente (il ne peut pas faire partie d'une unité d'oeuvre).

Si l'application insère ou obtient les messages dans le cadre d'une unité de travail, et que l'application décide alors d'en revenir à l'unité de travail, les informations de groupe et de segment sont restaurées à la valeur qu'elle avait précédemment:

- Les informations associées à l'appel MQPUT sont restaurées à la valeur qu'elles avaient avant le premier appel MQPUT réussi pour ce descripteur de file d'attente dans l'unité de travail en cours.
- Les informations associées à l'appel MQGET sont restaurées à la valeur qu'elles avaient avant le premier appel MQGET réussi pour ce descripteur de file d'attente dans l'unité d'oeuvre en cours.

Les files d'attente qui ont été mises à jour par l'application après le démarrage de l'unité de travail, mais en dehors de la portée de l'unité de travail, ne voient pas leurs informations de groupe et de segment restaurées si l'unité de travail est annulée.

La restauration des informations de groupe et de segment à sa valeur précédente lors de l'annulation d'une unité de travail permet à l'application de répartir un grand groupe de messages ou un grand message logique composé de plusieurs segments sur plusieurs unités de travail, et de redémarrer au point approprié du groupe de messages ou du message logique si l'une des unités de travail échoue.

L'utilisation de plusieurs unités d'oeuvre peut s'avérer avantageuse si le gestionnaire de files d'attente local ne dispose que d'une mémoire de file d'attente limitée. Toutefois, l'application doit conserver suffisamment d'informations pour pouvoir redémarrer l'insertion ou l'obtention de messages au point approprié en cas de défaillance du système.

Pour plus de détails sur le redémarrage au point approprié après une défaillance du système, voir l'option PMLOGO décrite dans PMOPT (entier signé à 10 chiffres) et l'option GMLOGO décrite dans GMOPT (entier signé à 10 chiffres).

Les autres remarques d'utilisation s'appliquent uniquement lorsque le gestionnaire de files d'attente coordonne les unités de travail:

- Une unité de travail a la même portée qu'un descripteur de connexion. Tous les appels IBM MQ qui affectent une unité de travail particulière doivent être exécutés à l'aide du même descripteur de

connexion. Les appels émis à l'aide d'un descripteur de connexion différent (par exemple, les appels émis par une autre application) affectent une unité d'oeuvre différente. Voir [HCONN \(entier signé à 10 chiffres\)-output](#) pour plus d'informations sur la portée des descripteurs de connexion.

- Seuls les messages insérés ou extraits dans le cadre de l'unité d'oeuvre en cours sont affectés par cet appel.
- Une application à exécution longue qui émet des appels MQGET, MQPUT ou MQPUT1 dans une unité de travail, mais qui n'émet jamais d'appel de validation ou d'annulation, peut remplir les files d'attente avec des messages qui ne sont pas disponibles pour d'autres applications. Pour éviter cette possibilité, l'administrateur doit définir l'attribut de gestionnaire de files d'attente **MaxUncommittedMsgs** sur une valeur suffisamment faible pour empêcher les applications en boucle indéfinie de remplir les files d'attente, mais suffisamment élevée pour permettre aux applications de messagerie attendues de fonctionner correctement.

## Paramètres

L'appel MQSETMP comporte les paramètres suivants:

### **HCONN (entier signé à 10 chiffres)-entrée**

Ce descripteur représente la connexion au gestionnaire de files d'attente.

La valeur doit correspondre au descripteur de connexion utilisé pour créer le descripteur de message spécifié dans le paramètre **HMSG** .

Si le descripteur de message a été créé à l'aide de HCUNAS, une connexion valide doit être établie sur l'unité d'exécution définissant une propriété du descripteur de message, sinon l'appel échoue avec le code anomalie RC2009 .

### **HMSG (entier signé à 20 chiffres)-entrée**

Il s'agit du descripteur de message à modifier. La valeur a été renvoyée par un appel MQCRTMH précédent.

### **SETOPT (MQSMPO)-entrée**

Contrôle la façon dont les propriétés de message sont définies.

Cette structure permet aux applications de spécifier des options qui contrôlent la manière dont les propriétés de message sont définies. La structure est un paramètre d'entrée dans l'appel MQSETMP. Pour plus d'informations, voir [MQSMPO](#) .

### **PRNAME (MQCHARV)-entrée**

Il s'agit du nom de la propriété à définir.

Pour plus d'informations sur l'utilisation des noms de propriété, voir [Noms de propriété](#) et [Restrictions relatives aux noms de propriété](#) .

### **PRPDSC (MQPD)-entrée/sortie**

Cette structure est utilisée pour définir les attributs d'une propriété, notamment:

- que se passe-t-il si la propriété n'est pas prise en charge
- quel est le contexte de message auquel appartient la propriété
- les messages dans lequel la propriété est copiée au fur et à mesure qu'elle est transmise

Pour plus d'informations sur cette structure, voir [MQPD](#) .

### **TYPE (entier signé à 10 chiffres)-entrée**

Type de données de la propriété en cours de définition. Il peut s'agir de l'un des éléments suivants :

#### **Connaissance de type**

Une valeur booléenne. *ValueLength* doit être 4.

**TYPBST**

Chaîne d'octets. *ValueLength* doit être supérieur ou égal à zéro.

**TYPI8**

Entier signé de 8 bits. *ValueLength* doit être 1.

**TYPI16**

Entier signé de 16 bits. *ValueLength* doit être 2.

**TYPI32**

Entier signé 32 bits. *ValueLength* doit être 4.

**TYPI64**

Entier signé 64 bits. *ValueLength* doit être 8.

**TYPF32**

Nombre à virgule flottante 32 bits. *ValueLength* doit être 4.

**TYPF64**

Nombre à virgule flottante de 64 bits. *ValueLength* doit être 8.

**TYPSTR**

Chaîne de caractères. *ValueLength* doit être supérieur ou égal à zéro ou la valeur spéciale VLNULL.

**TYPNUL**

La propriété existe mais a une valeur nulle. *ValueLength* doit être égal à zéro.

**VALLEN (entier signé à 10 chiffres)-entrée**

Longueur en octets de la valeur de propriété dans le paramètre *Valeur* .

La valeur zéro est valide uniquement pour les valeurs null ou pour les chaînes ou les chaînes d'octets. La valeur zéro indique que la propriété existe mais que la valeur ne contient ni caractères ni octets.

La valeur doit être supérieure ou égale à zéro ou à la valeur spéciale suivante si TYPSTR est défini pour le paramètre *Type* :

**VLNULL**

La valeur est délimitée par la première valeur nulle rencontrée dans la chaîne. La valeur null n'est pas incluse dans la chaîne. Cette valeur n'est pas valide si TYPSTR n'est pas également défini.

Remarque: Le caractère null utilisé pour terminer une chaîne si VLNULL est défini est une valeur null du jeu de caractères de la valeur.

**VALUE (chaîne de 1 octet x VALLEN)-entrée**

Valeur de la propriété à définir. La mémoire tampon doit être alignée sur une limite appropriée à la nature des données de la valeur.

Dans le langage de programmation C, le paramètre est déclaré comme un pointeur vers vide ; l'adresse de tout type de données peut être spécifiée comme paramètre.

Si *ValueLength* est égal à zéro, *Valeur* n'est pas référencé. Dans ce cas, l'adresse de paramètre transmise par les programmes écrits en assembleur C ou System/390 peut être null.

**CMPCOD (entier signé à 10 chiffres)-sortie**

Code achèvement ; il s'agit d'un des codes suivants :

**CCOK**

Achèvement réussi.

**CCFAIL**

Echec de l'appel.

**REASON (entier signé à 10 chiffres)-sortie**

Code anomalie qualifiant *CMPCOD*.

Si *CMPCOD* est CCOK:

**RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CMPCOD* est CCWARN:

**RC2421**

(2421, X'0975') Un dossier MQRFH2 contenant des propriétés n'a pas pu être analysé.

Si *CMPCOD* est CCFAIL:

**RC2204**

(2204, X'089C') Carte non disponible.

**RC2130**

(2130, X'852') Impossible de charger le module de service d'adaptateur.

**RC2157**

(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

**RC2004**

(2004, X'07D4') Paramètre de valeur non valide.

**RC2005**

(2005, X'07D5') Paramètre de longueur de valeur non valide.

**RC2219**

(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

**RC2460**

(2460, X'099C') Le pointeur de descripteur de message n'est pas valide.

**RC2499**

(2499, X'09C3') Descripteur de message déjà utilisé.

**RC2046**

(2046, X'07FE') Options non valides ou non cohérentes.

**RC2482**

(2482, X'09B2') Structure de descripteur de propriété non valide.

**RC2442**

(2442, X'098A') Nom de propriété non valide.

**RC2473**

(2473, X'09A9') Type de données de propriété non valide.

**RC2472**

(2472, X'09A8') Erreur de format numérique détectée dans les données de valeur.

**RC2463**

(2463, X'099F') La structure des options de propriété de message n'est pas valide.

**RC2111**

(2111, X'083F') Identificateur de jeu de caractères codés de nom de propriété non valide.

**RC2071**

(2071, X'817') Mémoire disponible insuffisante.

**RC2195**

(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations, voir [«Codes retour pour IBM i \(ILE RPG\)»](#), à la page 1498.

**Déclaration RPG**

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQSETMP(HCONN : HMSG : SETOPT :
                          PRNAME : PRPDSC :
                          TYPE : VALLEN : VALUE :
                          CMPCOD : REASON)

```



La définition de prototype pour l'appel est la suivante:

```
DMQSETMP          PR                EXTPROC('MQSETMP')
D* Connection handle
D HCONN           10I 0 VALUE
D* Message handle
D HMSG            10I 0 VALUE
D* Options that control the action of MQSETMP
D SETOPT          20A
D* Property name
D PRNAME          32A
D* Property descriptor
D PRPDSC          24A
D* Property data type
D TYPE            10I 0 VALUE
D* Length of the Value area
D VALLEN          10I 0 VALUE
D* Property value
D VALUE           *   VALUE
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CompCode
D REASON          10I 0
```

## MQSTAT (Extraire les informations de statut) sur IBM i

Utilisez l'appel MQSTAT pour extraire les informations de statut. Le type des informations de statut renvoyées est déterminé par la valeur STYPE spécifiée dans l'appel.

- [«Syntaxe», à la page 1425](#)
- [«Remarques sur l'utilisation», à la page 1425](#)
- [«Paramètres», à la page 1425](#)
- [«Déclaration RPG», à la page 1427](#)

### Syntaxe

(HCONN, STYPE, STAT, CMPCOD, REASON) MQSTAT

### Remarques sur l'utilisation

1. Un appel à MQSTAT spécifiant un type de STATAPT renvoie des informations sur les opérations MQPUT et MQPUT1 asynchrones précédentes. La structure MQSTAT transmise lors de l'appel est terminée avec les premières informations d'avertissement ou d'erreur asynchrone enregistrées pour cette connexion. Si d'autres erreurs ou avertissements suivent le premier, ils ne modifient normalement pas ces valeurs. Toutefois, si une erreur se produit avec le code achèvement CCWARN, un échec ultérieur avec le code achèvement CCFAIL est renvoyé à la place.
2. Si aucune erreur ne s'est produite depuis l'établissement de la connexion ou depuis le dernier appel à MQSTAT, un CMPCOD de CCOK et un MOTIF de RCNONE sont renvoyés.
3. Le nombre d'appels asynchrones traités sous le descripteur de connexion est renvoyé à l'aide de trois compteurs: STSPSC, STSPWC et STSPFC. Ces compteurs sont incrémentés par le gestionnaire de files d'attente chaque fois qu'une opération asynchrone est traitée avec succès, a un avertissement ou échoue (notez qu'à des fins de comptabilité, une insertion dans une liste de distribution compte une fois par file d'attente de destination plutôt qu'une fois par liste de distribution).
4. Un appel réussi à MQSTAT entraîne la réinitialisation des informations d'erreur ou des comptages précédents.

### Paramètres

L'appel MQSTAT comporte les paramètres suivants:

### **Hconn (MQHCONN)-entrée**

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

### **STYPE (entier signé à 10 chiffres)-entrée**

Type d'informations de statut demandées. La seule valeur valide est:

#### **STATAPT**

Renvoie des informations sur les opérations d'insertion asynchrone précédentes.

### **STS (MQSTS)-entrée/sortie**

Structure des informations de statut. Pour plus d'informations, voir «[MQSTS \(structure de génération de rapports de statut\) sous IBM i](#)», à la page 1291.

### **CMPCOD (entier signé à 10 chiffres)-sortie**

Code achèvement ; il s'agit d'un des codes suivants :

#### **CCOK**

Achèvement réussi.

#### **CCFAIL**

Echec de l'appel.

### **REASON (entier signé à 10 chiffres)-sortie**

Code anomalie qualifiant *CMPCOD*.

Si *CMPCOD* est CCOK:

#### **RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CMPCOD* est CCFAIL:

#### **RC2374**

(2374, X' 946') Echec de l'exit API

#### **RC2183**

(2183, X'887') Impossible de charger l'exit API.

#### **RC2219**

(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

#### **RC2009**

(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

#### **RC2203**

(2203, X'89B') Connexion en cours d'arrêt.

#### **RC2018**

(2018, X'7E2') Descripteur de connexion non valide.

#### **RC2162**

(2162, X'872') Arrêt du gestionnaire de files d'attente

#### **RC2102**

(2102, X'836') Ressources système disponibles insuffisantes.

#### **RC2430**

(2430, X'97E') Erreur avec le type MQSTAT.

#### **RC2071**

(2071, X'817') Mémoire disponible insuffisante.

#### **RC2424**

(2424, X' 978') Erreur avec la structure MQSTS

#### **RC2195**

(2195, X'893') Une erreur inattendue s'est produite.

## RC2298

(2298, X'8FA') La fonction demandée n'est pas disponible dans l'environnement en cours.

Pour plus d'informations sur ces codes, voir:

- [Messages et codes anomalie](#)

## Déclaration RPG

```
C*.. 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7
C          CALLP          MQSTAT(HCONN : ETYPE : ERR :
C                                CMPCOD : REASON)
```

La définition de prototype pour l'appel est la suivante:

```
D.. 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7
DMQSTAT          PR          EXTPROC('MQSTAT')
D* Connection handle
D HCONN          10I 0 VALUE
D* Status information type
D STYPE          10I 0 VALUE
D* Status information
D STATUS          296A
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CompCode
D REASON          10I 0
```

## IBM i MQSUB (Register Subscription) sur IBM i

L'appel MQSUB enregistre l'abonnement des applications à une rubrique particulière.

- «Syntaxe», à la page [1427](#)
- «Remarques sur l'utilisation», à la page [1427](#)
- «Paramètres», à la page [1429](#)
- «Déclaration RPG», à la page [1432](#)

## Syntaxe

MQSUB (*HCONN*, *SUBDSC*, *HOBJ*, *HSUB*, *CMPCOD*, *REASON*)

## Remarques sur l'utilisation

- L'abonnement est effectué à une rubrique, nommée à l'aide du nom abrégé d'un objet de rubrique prédéfini, du nom complet de la chaîne de rubrique ou de la concaténation de deux parties, comme décrit dans la rubrique [Combinaison de chaînes de rubrique](#).
- Le gestionnaire de files d'attente effectue des contrôles de sécurité lorsqu'un appel MQSUB est émis pour vérifier que l'ID utilisateur sous lequel l'application s'exécute dispose du niveau de droits approprié avant que l'accès ne soit autorisé. L'objet de rubrique approprié est localisé soit par un nom abrégé fourni dans l'appel, soit par l'objet de nom abrégé le plus proche dans la hiérarchie de rubriques trouvé si un nom long est fourni. Une vérification des droits est effectuée sur cet objet de rubrique pour s'assurer que les droits d'abonnement sont définis et sur la file d'attente de destination pour s'assurer que les droits de sortie sont définis. Si l'option SDMAN est utilisée, cela signifie qu'une vérification des droits d'accès est effectuée sur le nom de la file d'attente gérée associée à cet objet de rubrique, et si une file d'attente non gérée est fournie, cela signifie qu'une vérification des droits d'accès est effectuée sur la file d'attente représentée par le paramètre **HOBJ**.
- Le *HOBJ* renvoyé dans l'appel MQSUB lorsque l'option SOMAN est utilisée peut être interrogé afin de trouver des attributs tels que le seuil d'annulation et le nom de remise en file d'attente d'annulation excessive. Vous pouvez également consulter le nom de la file d'attente gérée, mais vous ne devez pas tenter d'ouvrir directement cette file d'attente.

- Les abonnements peuvent être regroupés, ce qui permet de distribuer une seule publication au groupe d'abonnements même si plusieurs groupes correspondent à la publication. Les abonnements sont regroupés à l'aide de l'option SOGRP et, pour regrouper les abonnements, ils doivent:
  - utiliser la même file d'attente nommée (qui n'utilise pas l'option SOMAN) sur le même gestionnaire de files d'attente-représenté par le paramètre **HOBJ** sur l'appel MQSUB
  - partager le même *SDCID*
  - être du même *SDSL*

Ces attributs définissent l'ensemble des abonnements considérés comme faisant partie du groupe et sont également les attributs qui ne peuvent pas être modifiés si un abonnement est groupé. L'altération des résultats *SDSL* dans RC2512 et l'altération de tous les autres (qui peuvent être modifiés si un abonnement n'est pas groupé) se traduit par RC2515.

- Les zones du MQSD sont renseignées en cas de retour d'un appel MQSUB qui utilise l'option SORES. Le MQSD renvoyé peut être transmis directement dans un appel MQSUB qui utilise l'option SOALT avec les modifications que vous devez apporter à l'abonnement appliqué au MQSD. Certaines zones comportent des considérations spéciales, comme indiqué dans le tableau.

Nom de zone dans MQSD	Des considérations spéciales
Options d'accès ou de création	Aucune de ces options n'est définie lors du retour de l'appel MQSUB. Si vous réutilisez ultérieurement le MQSD dans un appel MQSUB, l'option dont vous avez besoin doit être explicitement définie.
Options de durabilité, Options de destination, Options d'enregistrement et Options de caractère générique	Ces options seront définies comme il convient
Options de publication	Ces options seront définies selon les besoins, à l'exception de SONEWP qui s'applique uniquement à SOCRE.
Autres options	Ces options sont inchangées en cas de retour d'un appel MQSUB. Ils contrôlent la façon dont l'appel d'API est émis et ne sont pas stockés avec l'abonnement. Ils doivent être définis comme requis lors de tout appel MQSUB ultérieur réutilisant le MQSD.
ObjectName	Cette zone d'entrée uniquement n'est pas modifiée en cas de retour d'un appel MQSUB.
ObjectString	Cette zone d'entrée uniquement n'est pas modifiée en cas de retour d'un appel MQSUB. Le nom de rubrique complet utilisé est renvoyé dans la zone <i>SDRO</i> , si une mémoire tampon est fournie.
ID AlternateUser et ID AlternateSecurity	Ces zones d'entrée uniquement sont inchangées en cas de retour d'un appel MQSUB. Ils contrôlent la façon dont l'appel d'API est émis et ne sont pas stockés avec l'abonnement. Ils doivent être définis comme requis lors de tout appel MQSUB ultérieur réutilisant le MQSD.
SubExpiry	En cas de retour d'un appel MQSUB à l'aide de l'option SORES, cette zone sera définie sur l'expiration d'origine de l'abonnement et non sur l'heure d'expiration restante. Si vous réutilisez ensuite le MQSD dans un appel MQSUB à l'aide de l'option SOALT, vous réinitialisez l'expiration de l'abonnement pour recommencer à décompter.
SubName	Cette zone est une zone d'entrée dans un appel MQSUB et n'est pas modifiée dans la sortie.

Tableau 752. Sortie MQSD de MQSUB (suite)	
Nom de zone dans MQSD	Des considérations spéciales
SubUserData et SelectionString	<p>Ces zones de longueur variable sont renvoyées en sortie d'un appel MQSUB à l'aide de l'option SORES, si une mémoire tampon est fournie, ainsi qu'une longueur de mémoire tampon positive dans VCHRP. Si aucune mémoire tampon n'est fournie, seule la longueur est renvoyée dans la zone VCHRL de MQCHARV. If la mémoire tampon fournie est inférieure à l'espace requis pour renvoyer la zone, seuls VCHRP octets sont renvoyés dans la mémoire tampon fournie.</p> <p>Si vous réutilisez ultérieurement le MQSD dans un appel MQSUB à l'aide de l'option SOALT et qu'une mémoire tampon n'est pas fournie, mais qu'une valeur VCHRL différente de zéro est fournie, si cette longueur correspond à la longueur existante de la zone, aucune modification n'est apportée à la zone.</p>
ID SubCorrelet jeton PubAccounting	<p>Si vous n'utilisez pas SOSCID, SDCID sera généré par le gestionnaire de files d'attente. Si vous n'utilisez pas SOSETI, le SDACC sera généré par le gestionnaire de files d'attente.</p> <p>Ces zones sont renvoyées dans le MQSD à partir d'un appel MQSUB à l'aide de l'option SORES. Si elles sont générées par le gestionnaire de files d'attente, la valeur générée est renvoyée lors d'un appel MQSUB à l'aide de l'option SOCRE ou SOALT.</p>
PubPriority, SubLevel & PubApplIdentityData	Ces zones seront renvoyées dans le MQSD.
Chaîne ResObject	Cette zone de sortie uniquement est renvoyée dans le MQSD si une mémoire tampon est fournie.

## Paramètres

L'appel MQSUB comporte les paramètres suivants:

### HCONN (entier signé à 10 chiffres)-entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de HCONN a été renvoyée par un appel MQCONN ou MQCONNX précédent.

### SUBDSC (MQSD)-entrée-sortie

Il s'agit d'une structure qui identifie l'objet utilisé qui est enregistré par l'application. Pour plus d'informations, voir «MQSD (descripteur d'abonnement) sous IBM i», à la page 1272.

### HOBJ (entier signé à 10 chiffres)-entrée / sortie

Ce descripteur représente l'accès qui a été établi pour obtenir les messages envoyés à cet abonnement. Ces messages peuvent être stockés dans une file d'attente spécifique ou le gestionnaire de files d'attente peut être invité à gérer leur stockage sans avoir besoin d'une file d'attente spécifique.

Descripteur d'objet.

Si une file d'attente spécifique doit être utilisée, elle doit être associée à l'abonnement au moment de la création. Cela peut se faire de deux façons:

- En fournissant ce descripteur lors de l'appel de MQSUB avec l'option SDCRT. Si ce descripteur est fourni en tant que paramètre d'entrée dans l'appel, il doit s'agir d'un descripteur d'objet valide renvoyé par un appel MQOPEN précédent d'une file d'attente utilisant au moins l'une des options OOINP\*, OOOOUT (si une file d'attente éloignée, par exemple) ou OOBROW. Si tel n'est pas le cas, l'appel échoue avec RC2019. Il ne peut pas s'agir d'un descripteur d'objet dans une file d'attente alias qui se résout en un objet de rubrique. Si tel est le cas, l'appel échoue avec RC2019
- En utilisant la commande DEFINE SUB MQSC et en indiquant cette commande avec le nom d'un objet file d'attente.

Si le gestionnaire de files d'attente doit gérer le stockage des messages envoyés à cet abonnement, vous devez l'indiquer lors de la création de l'abonnement, en utilisant l'option SOMAN et en définissant la valeur de paramètre sur HONONE. Le gestionnaire de files d'attente renvoie le descripteur en tant que paramètre de sortie sur l'appel et le descripteur renvoyé est appelé descripteur géré. Si HONONE est spécifié et que SOMAN n'est pas également spécifié, l'appel échoue avec RC2019.

Un descripteur géré renvoyé par le gestionnaire de files d'attente peut être utilisé sur un appel MQGET ou MQCB, avec ou sans options de navigation, sur un appel MQINQ ou sur MQCLOSE. Il ne peut pas être utilisé sur MQPUT, MQSET ou sur un MQSUB suivant ; la tentative échoue avec RC2039 pour MQPUT, RC2040 pour MQSET ou RC2038 pour MQSUB.

Si l'option SORES dans la zone *OPTS* de la structure MQSD est utilisée pour reprendre cet abonnement, le descripteur peut être renvoyé à l'application dans ce paramètre si HONONE est spécifié. Vous pouvez l'utiliser, que l'abonnement utilise un descripteur géré ou non. Il peut être utile pour les abonnements créés à l'aide de DEFINE SUB si vous souhaitez que le descripteur de la file d'attente d'abonnement soit défini dans la commande DEFINE SUB. Dans le cas où un abonnement créé administrativement est repris, la file d'attente est ouverte avec OOINPQ et OOBW. Si d'autres options sont nécessaires, l'application doit ouvrir la file d'attente d'abonnement de manière explicite et fournir le descripteur d'objet lors de l'appel. En cas de problème lors de l'ouverture de la file d'attente, l'appel échoue avec RC2522. Si *HOBJ* est fourni, il doit être équivalent à *HOBJ* dans l'appel MQSUB d'origine. Cela signifie que si un descripteur d'objet renvoyé par un appel MQOPEN est fourni, le descripteur doit se trouver dans la même file d'attente que celle utilisée précédemment, sinon l'appel échoue avec RC2019.

Si cet abonnement est modifié, en utilisant l'option SOALT dans la zone *OPTS* de la structure MQSD, un autre *HOBJ* peut être fourni. Toutes les publications qui ont été distribuées à la file d'attente précédemment identifiée à l'aide de ce paramètre restent dans cette file d'attente et il incombe à l'application d'extraire ces messages si le paramètre **HOBJ** représente désormais une file d'attente différente.

L'utilisation de ce paramètre avec différentes options d'abonnement est résumée dans le tableau suivant:

Options	HOBJ	Description
SOCRT + SOMAN	Ignoré en entrée	Crée un abonnement avec un stockage de messages géré par le gestionnaire de files d'attente.
SOCRT	Descripteur d'objet valide	Crée un abonnement fournissant une file d'attente spécifique comme destination des messages.
SORES	HONONE	Reprend un abonnement créé précédemment (géré ou non) et demande au gestionnaire de files d'attente de renvoyer le descripteur d'objet à utiliser par l'application.
SORES	Descripteur d'objet valide, correspondant	Reprend un abonnement créé précédemment qui utilise une file d'attente spécifique comme destination des messages et utilise un descripteur d'objet avec des options d'ouverture spécifiques.
SOALT + SOMAN	HONONE	Modifie un abonnement existant qui utilisait auparavant une file d'attente spécifique à gérer.
SOALT	Descripteur d'objet valide	Modifie un abonnement existant pour utiliser une file d'attente spécifique (provenant d'une file d'attente gérée ou d'une file d'attente spécifique différente).

Qu'il ait été fourni ou renvoyé, *HOB*J doit être spécifié sur les appels MQGET suivants dont vous avez besoin pour recevoir les publications.

Le descripteur *HOB*J cesse d'être valide lorsque l'appel MQCLOSE est émis dessus ou lorsque l'unité de traitement qui définit la portée du descripteur s'arrête. La portée du descripteur d'objet renvoyé est identique à celle du descripteur de connexion spécifié dans l'appel. Pour plus d'informations sur la portée du descripteur, voir [HCONN](#) . Une exception MQCLOSE du descripteur *HOB*J n'a aucun effet sur le descripteur *HSUB* .

### **HSUB (entier signé à 10 chiffres)-sortie**

Ce descripteur représente l'abonnement qui a été effectué. Il peut être utilisé pour deux autres opérations:

- Il peut être utilisé lors d'un appel MQSUBRQ ultérieur pour demander l'envoi de publications lorsque l'option SOPUBR a été utilisée lors de l'abonnement.
- Il peut être utilisé lors d'un appel MQCLOSE ultérieur pour supprimer l'abonnement qui a été effectué. Le descripteur *HSUB* cesse d'être valide lorsque l'appel MQCLOSE est émis ou lorsque l'unité de traitement qui définit la portée du descripteur s'arrête. La portée du descripteur d'objet renvoyé est identique à celle du descripteur de connexion spécifié dans l'appel. Une exception MQCLOSE du descripteur *HSUB* n'a aucun effet sur le descripteur *HOB*J .

Ce descripteur ne peut pas être transmis à un appel MQGET ou MQCB. Vous devez utiliser le paramètre **HOB**J . La transmission de ce descripteur à tout autre résultat d'appel IBM MQ dans RC2019.

### **CMPCOD (entier signé à 10 chiffres)-sortie**

Code achèvement ; il s'agit d'un des codes suivants :

#### **CCOK**

Achèvement réussi

#### **CCWARN**

Avertissement (achèvement partiel)

#### **CCFAIL**

Echec de l'appel

### **REASON (entier signé à 10 chiffres)-sortie**

Code anomalie qualifiant *CMPCOD*.

Si *CMPCOD* est CCOK:

#### **RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CMPCOD* est CCFAIL:

#### **RC2019**

(2019 X'07E3') Descripteur d'objet non valide

#### **RC2046**

(2046 X'07FE') Options non valides ou non cohérentes

#### **RC2085**

(2085 X'0825') L'objet identifié est introuvable

#### **RC2161**

(2161 X'0871') Mise au repos du gestionnaire de files d'attente

#### **RC2298**

(2298 X'08FA') Fonction non prise en charge.

#### **RC2424**

(2424 X'0978') Descripteur d'abonnement (MQSD) non valide

#### **RC2425**

(2441 X' 979') Chaîne de rubrique non valide

**RC2428**

(2428 X'097C') Le nom d'abonnement spécifié ne correspond pas aux abonnements existants

**RC2429**

(2429 X'097D') Le nom d'abonnement existe et est utilisé par une autre application

**RC2431**

(2431 X'097F') Zone de données SubUser non valide

**RC2432**

(2432 X'0980') L'abonnement existe

**RC2434**

(2434 X'0982') Le nom de l'abonnement correspond à l'abonnement existant

**RC2440**

(2440 X'0988') Zone SubName non valide

**RC2441**

(2441 X'0989') Zone Objectstring non valide

**RC2435**

(2435 X'0983') L'attribut ne peut pas être modifié à l'aide de SDALT ou l'abonnement a été créé avec SDIMM.

**RC2436**

(2436 X'0984') L'option SODUR n'est pas valide

**RC2459**

(2459, X'99B') Erreur de syntaxe de la chaîne de sélection.

**RC2503**

(2503 X'09C7') Les appels MQSUB sont actuellement bloqués pour les rubriques auxquelles ils sont abonnés.

**RC2519**

(2519, X'9D7') La chaîne de sélection n'est pas telle que spécifiée dans la description de l'utilisation d'une structure MQCHARV.

**RC2551**

(2551, X'9F7') La chaîne de sélection spécifiée n'est pas disponible.

## Déclaration RPG

```

C*.1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQSUB(HCONN : SUBDSC : HOBJ :
C          HSUB : CMPCOD : REASON)

```

La définition de prototype pour l'appel est la suivante:

```

D*.1.....2.....3.....4.....5.....6.....7..
DMQSUB          PR          EXTPROC('MQSUB')
D* Connection handle
D HCONN          10I 0 VALUE
D* Subscription descriptor
D SUBDSC          400A
D* Object handle for queue
D HOBJ          10I 0
D* Subscription object handle
D HSUB          10I 0
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CompCode
D REASON          10I 0

```

## MQSUBRQ (demande d'abonnement) sur IBM i

L'appel MQSUBRQ effectue une demande sur un abonnement.



- «Syntaxe», à la page 1433
- «Remarques sur l'utilisation», à la page 1433
- «Paramètres», à la page 1433
- «Déclaration RPG», à la page 1434

## Syntaxe

*MQSUBRQ (HCONN, HSUB, ACTION, SUBROPT, CMPCOD, REASON)*

## Remarques sur l'utilisation

Les remarques d'utilisation suivantes s'appliquent à l'utilisation de SRAPUB:

1. Si cette instruction aboutit, les publications conservées correspondant à l'abonnement spécifié ont été envoyées à l'abonnement et peuvent être reçues à l'aide de MQGET ou de MQCB à l'aide du HOBJ renvoyé sur l'instruction MQSUB d'origine qui a créé l'abonnement.
2. Si la rubrique à laquelle souscrit l'instruction MQSUB d'origine qui a créé l'abonnement contient un caractère générique, plusieurs publications conservées peuvent être envoyées. Le nombre de publications envoyées à la suite de cet appel est enregistré dans la zone *SRNMP* de la structure *SBROPT*.
3. Si cette instruction se termine avec le code anomalie RC2437, cela signifie qu'aucune publication n'a été conservée pour la rubrique spécifiée.
4. Si cette instruction se termine avec le code anomalie RC2525 ou RC2526, des publications sont actuellement conservées pour la rubrique spécifiée, mais une erreur s'est produite, ce qui signifie qu'elles n'ont pas pu être distribuées.
5. L'application doit disposer d'un abonnement en cours à la rubrique pour pouvoir effectuer cet appel. Si l'abonnement a été effectué dans une instance précédente de l'application et qu'un descripteur valide de l'abonnement n'est pas disponible, l'application doit d'abord appeler MQSUB avec l'option SORES afin d'obtenir un descripteur pour l'utiliser dans cet appel.
6. Les publications sont envoyées à la destination enregistrée pour être utilisée avec l'abonnement en cours de cette application. Si les publications doivent être envoyées ailleurs, l'abonnement doit d'abord être modifié à l'aide de l'appel MQSUB avec l'option SOALT.

## Paramètres

L'appel MQSUBRQ comporte les paramètres suivants:

### **HCONN (entier signé à 10 chiffres)-entrée**

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *HCONN* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS, l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour *HCONN*:

#### **HCDEFH**

Descripteur de connexion par défaut.

### **HSUB (entier signé à 10 chiffres)-entrée**

Ce descripteur représente l'abonnement pour lequel une mise à jour doit être demandée. La valeur de *HSUB* a été renvoyée à partir d'un appel MQSUB précédent.

### **ACTION (entier signé à 10 chiffres)-entrée**

Ce paramètre contrôle l'action particulière demandée sur l'abonnement. Un (et un seul) des éléments suivants doit être spécifié:

## SRAPUB

Cette action demande l'envoi d'une publication de mise à jour pour la rubrique spécifiée. Cette option est normalement utilisée si l'abonné a spécifié l'option SOPUBR sur l'appel MQSUB lorsqu'il a effectué l'abonnement. Si le gestionnaire de files d'attente possède une publication conservée pour la rubrique, celle-ci est envoyée à l'abonné. Si ce n'est pas le cas, l'appel échoue. Si une application reçoit une publication qui a été conservée, elle est indiquée par la propriété de message MQIsRetained de cette publication.

Etant donné que la rubrique de l'abonnement existant représenté par le paramètre **HSUB** peut contenir des caractères génériques, l'abonné peut recevoir plusieurs publications conservées.

## SBROPT (MQSRO)-entrée/sortie

Ces options contrôlent l'action de MQSUBRQ. Pour plus de détails, voir «MQSRO-Options de demande d'abonnement», à la page 616 .

## CMPCOD (entier signé à 10 chiffres)-sortie

Code achèvement ; il s'agit d'un des codes suivants :

### CCOK

Achèvement réussi

### CCWARN

Avertissement (achèvement partiel)

### CCFAIL

Echec de l'appel

## Motif (entier signé à 10 chiffres)-sortie

Code anomalie qualifiant *CMPCOD*.

Si *CMPCOD* est CCOK:

### RCNONE

(0, X'000') Aucun code anomalie à signaler.

Si *CMPCOD* est CCFAIL:

### RC2298

2298 (X'08FA') La fonction demandée n'est pas disponible dans l'environnement en cours.

### RC2437

2437 (X'0985') Aucune publication conservée n'est actuellement stockée pour cette rubrique.

### RC2046

2046 (X'07FE') Le paramètre ou la zone Options contient des options non valides ou une combinaison d'options non valides.

### RC2161

2161 (X'0871') Mise au repos du gestionnaire de files d'attente

### RC2438

2438 (X'0986') Sur l'appel MQSUBRQ, les options de demande d'abonnement MQSRO ne sont pas valides.

## Déclaration RPG

```
C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQSUBRQ(HCONN : HSUB : ACTION :
C          SBROPT : CMPCOD : REASON)
```

La définition de prototype pour l'appel est la suivante:

```
D*..1.....2.....3.....4.....5.....6.....7..
DMQSUBRQ      PR          EXTPROC('MQSUBRQ')
```

```

D* Connection handle
D HCONN                10I 0 VALUE
D* Subscription handle
D HSUB                10I 0 VALUE
D* Action requested on the subscription
D ACTION              10I 0 VALUE
D* Subscription Request Options
D SBROPT              16A
D* Completion code
D CMPCOD              10I 0
D* Reason code qualifying CompCode
D REASON              10I 0

```

## IBM i Attributs des objets sur IBM i

Cette collection de rubriques répertorie uniquement les objets IBM MQ qui peuvent faire l'objet d'un appel de fonction MQINQ et fournit des détails sur les attributs pouvant être recherchés et les sélecteurs à utiliser.

### Attributs des files d'attente

Utilisez ces informations pour en savoir plus sur les différents types de définition de file d'attente et sur les attributs pris en charge par chacun d'eux.

**Types de file d'attente:** Le gestionnaire de files d'attente prend en charge les types de définition de file d'attente suivants:

#### File d'attente locale

Il s'agit d'une file d'attente physique qui stocke les messages. La file d'attente existe sur le gestionnaire de files d'attente local.

Les applications connectées au gestionnaire de files d'attente local peuvent placer des messages dans des files d'attente de ce type et en supprimer. La valeur de l'attribut de file d'attente **QType** est QTLOC.

#### File d'attente partagée

Il s'agit d'une file d'attente physique qui stocke les messages. La file d'attente existe dans un référentiel partagé accessible à tous les gestionnaires de files d'attente appartenant au groupe de partage de files d'attente propriétaire du référentiel partagé.

Les applications connectées à un gestionnaire de files d'attente dans le groupe de partage de files d'attente peuvent placer des messages dans des files d'attente de ce type et en supprimer. Ces files d'attente sont en fait les mêmes que les files d'attente locales. La valeur de l'attribut de file d'attente **QType** est QTLOC.

- Les files d'attente partagées sont prises en charge uniquement sous z/OS.

#### File d'attente de cluster

Il s'agit d'une file d'attente physique qui stocke les messages. La file d'attente existe sur le gestionnaire de files d'attente local ou sur un ou plusieurs des gestionnaires de files d'attente appartenant au même cluster que le gestionnaire de files d'attente local.

Les applications connectées au gestionnaire de files d'attente local peuvent placer des messages dans des files d'attente de ce type, quel que soit l'emplacement de la file d'attente. Si une instance de la file d'attente existe sur le gestionnaire de files d'attente local, la file d'attente se comporte de la même manière qu'une file d'attente locale et les applications connectées au gestionnaire de files d'attente local peuvent supprimer des messages de la file d'attente. La valeur de l'attribut de file d'attente **QType** est QTCLUS.

#### File d'attente d'alias

Il ne s'agit pas d'une file d'attente physique ; il s'agit d'un autre nom pour une file d'attente locale. Le nom de la file d'attente locale constituant la résolution de l'alias fait partie de la définition de la file d'attente alias.

Les applications connectées au gestionnaire de files d'attente local peuvent placer des messages dans des files d'attente alias et en supprimer. Les messages sont placés dans la file d'attente locale dans laquelle l'alias est résolu et supprimés. La valeur de l'attribut de file d'attente **QType** est QTALS.

### File d'attente éloignée

Il ne s'agit pas d'une file d'attente physique-il s'agit de la définition locale d'une file d'attente qui existe sur un gestionnaire de files d'attente éloignées. La définition locale de la file d'attente éloignée contient des informations indiquant au gestionnaire de files d'attente local comment acheminer les messages vers le gestionnaire de files d'attente éloignées.

Les applications connectées au gestionnaire de files d'attente local peuvent placer des messages dans des files d'attente éloignées-les messages sont placés dans la file d'attente de transmission locale utilisée pour acheminer les messages vers le gestionnaire de files d'attente éloignées. Les applications ne peuvent pas supprimer les messages des files d'attente éloignées. La valeur de l'attribut de file d'attente **QType** est QTREM.

Une définition de file d'attente éloignée peut également être utilisée pour:

- Alias de file d'attente de réponses

Dans ce cas, le nom de la définition est le nom d'une file d'attente de réponse. Pour plus d'informations, voir [Définitions d'alias de file d'attente de réponse](#).

- Alias du gestionnaire de files d'attente

Dans ce cas, le nom de la définition est un alias pour un gestionnaire de files d'attente et non le nom d'une file d'attente. Pour plus d'informations, voir [Définitions d'alias de gestionnaire de files d'attente](#).

### File d'attente modèle

Il ne s'agit pas d'une file d'attente physique ; il s'agit d'un ensemble d'attributs de file d'attente à partir desquels une file d'attente locale peut être créée.

Les messages ne peuvent pas être stockés dans des files d'attente de ce type.

Certains attributs de file d'attente s'appliquent à tous les types de file d'attente ; d'autres s'appliquent uniquement à certains types de file d'attente. Les types de file d'attente auxquels s'applique un attribut sont indiqués par un "X" dans [Tableau 754](#), à la [page 1436](#) et les tables suivantes.

Le [Tableau 754](#), à la [page 1436](#) récapitule les attributs spécifiques aux files d'attente. Les attributs sont décrits par ordre alphabétique.

Les noms des attributs affichés dans la table sont ceux utilisés avec les appels MQINQ et MQSET. Lorsque des commandes MQSC sont utilisées pour définir, modifier ou afficher des attributs, des noms abrégés alternatifs sont utilisés ; voir [Commandes MQSC](#) pour plus de détails.

Dans le tableau suivant, les colonnes s'appliquent comme suit:

- La colonne des files d'attente locales s'applique également aux files d'attente partagées.
- La colonne des files d'attente modèles indique les attributs hérités par la file d'attente locale créée à partir de la file d'attente modèle.
- La colonne des files d'attente de cluster indique les attributs qui peuvent être renseignés lorsque la file d'attente de cluster est ouverte pour l'interrogation seule ou pour l'interrogation et la sortie. Si la file d'attente de cluster est ouverte pour l'interrogation plus une ou plusieurs entrées, parcourez ou définissez, la colonne des files d'attente locales s'applique à la place.

Attribut	Description	Locale	Modèle	Alias	Eloigné	Cluster
<a href="#">AlterationDate</a>	Date de la dernière modification de la définition	X		X	X	

Tableau 754. Attributs des files d'attente (suite)						
Attribut	Description	Locale	Modèle	Alias	Eloigné	Cluster
<u>AlterationTime</u>	Heure de la dernière modification de la définition	X		X	X	
<u>BackoutRequeueQName</u>	Nom de file d'attente de remise en file d'attente d'annulation excessive	X	X			
<u>BackoutThreshold</u>	Seuil d'annulation	X	X			
<u>BaseQName</u>	Nom de la file d'attente dans laquelle l'alias est résolu			X		
<u>ClusterChannelNom</u>	Nom du canal émetteur de cluster	✓	✓			
<u>ClusterName</u>	Nom du cluster auquel appartient la file d'attente	X		X	X	
<u>ClusterNameList</u>	Nom de l'objet liste de noms contenant les noms des clusters auxquels appartient la file d'attente	X		X	X	
<u>CreationDate</u>	Date de création de la file d'attente	X				
<u>CreationTime</u>	Heure de création de la file d'attente	X				
<u>CurrentQDepth</u>	Longueur actuelle de la file	X				
<u>DefBind</u>	Liaison par défaut	X		X	X	X
<u>DefinitionType</u>	Type de définition de file d'attente	X	X			
<u>DefInputOpenOption</u>	Option d'ouverture en entrée par défaut	X	X			
<u>DefPersistence</u>	Persistance par défaut	X	X	X	X	X
<u>DefPriority</u>	Priorité par défaut	X	X	X	X	X
<u>DistLists</u>	Prise en charge de la liste de diffusion	X	X			
<u>HardenGetBackout</u>	Indique s'il convient de conserver un nombre d'annulations précis	X	X			

Tableau 754. Attributs des files d'attente (suite)

Attribut	Description	Locale	Modèle	Alias	Eloigné	Cluster
<a href="#">InhibitGet</a>	Contrôle si les opérations d'extraction de la file d'attente sont autorisées	X	X	X		
<a href="#">InhibitPut</a>	Contrôle si les opérations d'insertion pour la file d'attente sont autorisées	X	X	X	X	X
<a href="#">InitiationQName</a>	Nom de la file d'attente d'initialisation	X	X			
<a href="#">Longueur maximale des messages</a>	Longueur maximale des messages (en octets)	X	X			
<a href="#">MaxQDepth</a>	Long. max. file d'attente	X	X			
<a href="#">MediaLog</a>	Identité de l'extension de journal la plus ancienne (ou du récepteur de journal le plus ancien sur IBM i) nécessaire pour la reprise sur incident lié au support d'une file d'attente spécifiée	✓	✓			
<a href="#">MsgDeliverySequence</a>	Séquence livraison messages	X	X			
<a href="#">OpenInputCount</a>	Nombre d'ouvertures pour la saisie	X				
<a href="#">OpenOutputCount</a>	Nombre d'ouvertures pour la sortie	X				
<a href="#">ProcessName</a>	Nom du processus	X	X			
<a href="#">QDepthHighEvent</a>	Contrôle si les événements Longueur élevée de la file d'attente sont générés	X	X			
<a href="#">QDepthHighLimit</a>	Limite supérieure de la longueur de la file d'attente	X	X			
<a href="#">QDepthLowEvent</a>	Contrôle si les événements Longueur faible de la file d'attente sont générés	X	X			

Tableau 754. Attributs des files d'attente (suite)

Attribut	Description	Locale	Modèle	Alias	Eloigné	Cluster
<u>QDepthLowLimit</u>	Limite inférieure de la longueur de la file d'attente	X	X			
<u>QDepthMaxEvent</u>	Contrôle si les événements File d'attente saturée sont générés	X	X			
<u>QDesc</u>	Description de file d'attente	X	X	X	X	X
<u>QName</u>	Nom de la file d'attente	X		X	X	X
<u>QServiceInterval</u>	Cible de l'intervalle de service de file d'attente	X	X			
<u>ÉvénementQServiceInterval</u>	Contrôle si les événements Intervalle de service élevé ou Intervalle de service OK sont générés	X	X			
<u>QType</u>	Type de file d'attente	X		X	X	X
<u>RemoteQMgrName</u>	Nom du gestionnaire de files d'attente éloignées				X	
<u>RemoteQName</u>	Nom de la file d'attente éloignée				X	
<u>RetentionInterval</u>	Intervalle de conservation	X	X			
<u>Portée</u>	Contrôle si une entrée de la file d'attente existe également dans un répertoire de cellule	X		X	X	
<u>Partage de file d'attente</u>	Partageabilité de la file d'attente	X	X			
<u>TriggerControl</u>	Contrôle du déclenchement	X	X			
<u>TriggerData</u>	Données de déclenchement	X	X			
<u>TriggerDepth</u>	Longueur de déclenchement	X	X			
<u>TriggerMsgPriority</u>	Priorité des messages de seuil des déclencheurs	X	X			

Attribut	Description	Locale	Modèle	Alias	Eloigné	Cluster
<u>TriggerType</u>	Type de déclenchement	X	X			
<u>Utilisation</u>	Utilisation de la file d'attente	X	X			
<u>XmitQName</u>	Nom de la file d'attente de transmission				X	

### IBM i **AlterationDate (chaîne de caractères de 12 octets) sur IBM i**

Date de la dernière modification de la définition.

Locale	Modèle	Alias	Eloigné	Cluster
X		X	X	

Il s'agit de la date de la dernière modification de la définition. Le format de la date est YYYY-MM-DD, complété par deux blancs de fin pour obtenir une longueur de 12 octets (par exemple, 1992-09-23↵↵, où ↵↵ représente deux caractères blancs).

Les valeurs de certains attributs (par exemple, *CurrentQDepth*) changent lorsque le gestionnaire de files d'attente fonctionne. Les modifications apportées à ces attributs n'affectent pas *AlterationDate*.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAALTD avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNDATE.

### IBM i **AlterationTime (chaîne de caractères de 8 octets) sur IBM i**

Heure de la dernière modification de la définition.

Locale	Modèle	Alias	Eloigné	Cluster
X		X	X	

Il s'agit de l'heure à laquelle la définition a été modifiée pour la dernière fois. Le format de l'heure est HH.MM.SS à l'aide de l'horloge au format 24 heures, avec un zéro non significatif si l'heure est inférieure à 10 (par exemple, 09.10.20). L'heure est l'heure locale.

Les valeurs de certains attributs (par exemple, *CurrentQDepth*) changent lorsque le gestionnaire de files d'attente fonctionne. Les modifications apportées à ces attributs n'affectent pas *AlterationTime*.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAALTT avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNTIME.

### IBM i **BackoutRequeueQName (chaîne de caractères de 48 octets) sur IBM i**

Nom de file d'attente de remise en file d'attente d'annulation trop élevé.



Tableau 757. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Les applications qui s'exécutent dans WebSphere Application Server et celles qui utilisent IBM MQ Application Server Facilities utilisent cet attribut pour déterminer l'emplacement des messages qui ont été annulés. Pour toutes les autres applications, en dehors de l'autorisation d'interrogation de sa valeur, le gestionnaire de files d'attente n'effectue aucune action en fonction de la valeur de l'attribut.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CABRQN avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNQN.

**IBM i BackoutThreshold (entier signé à 10 chiffres) sur IBM i**

Seuil d'annulation.

Tableau 758. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Les applications qui s'exécutent dans WebSphere Application Server et celles qui utilisent IBM MQ Application Server Facilities utilisent cet attribut pour déterminer si un message doit être annulé. Pour toutes les autres applications, en dehors de l'autorisation d'interrogation de sa valeur, le gestionnaire de files d'attente n'effectue aucune action en fonction de la valeur de l'attribut.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IABTHR avec l'appel MQINQ.

**IBM i BaseQName (chaîne de caractères de 48 octets) sur IBM i**

Nom de la file d'attente dans laquelle l'alias est résolu.

Tableau 759. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
		X		

Il s'agit du nom d'une file d'attente définie pour le gestionnaire de files d'attente local. (Pour plus d'informations sur les noms de file d'attente, voir la description de la zone *ODON* dans MQOD. La file d'attente est de l'un des types suivants:

**Intervalle QTLOC**

File d'attente locale.

**QTREM**

Définition locale d'une file d'attente éloignée.

**QTCLUS**

File d'attente de cluster.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CABASQ avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNQN.

**IBM i BaseType (structure de paramètre entier) sur IBM i**

Type d'objet auquel l'alias est résolu.

Tableau 760. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
		X		

Cet attribut peut adopter l'une des valeurs suivantes :

**File d'attente OTQ**

Le type d'objet de base est une file d'attente

**Mot de passe OTTOP**

Le type d'objet de base est une rubrique

**IBM i CFStrucName (chaîne de caractères de 12 octets) sous IBM i**

Nom de la structure d'unité de couplage.

Tableau 761. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Il s'agit du nom de la structure d'unité de couplage dans laquelle les messages de la file d'attente sont stockés. Le premier caractère du nom est compris entre A et Z, et les caractères restants sont compris entre A et Z, 0 et 9, ou à blanc.

Le nom complet de la structure dans l'unité de couplage est obtenu en suffixant la valeur de l'attribut de gestionnaire de files d'attente **QSGName** avec la valeur de l'attribut de file d'attente **CFStrucName**.

Cet attribut s'applique uniquement aux files d'attente partagées ; il est ignoré si *QSGDisp* ne possède pas la valeur QSGDSH.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CACFSN avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNCFSN.

**z/OS** Cet attribut est pris en charge uniquement sous z/OS.

**ClusterChannelNom (chaîne de caractères de 20 octets)**

ClusterChannelNom est le nom générique des canaux émetteurs de cluster qui utilisent cette file d'attente comme file d'attente de transmission. L'attribut indique quels canaux émetteurs de cluster envoient des messages à un canal récepteur de cluster à partir de cette file d'attente de transmission de cluster.

Tableau 762. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

La configuration du gestionnaire de files d'attente par défaut s'applique à tous les canaux émetteurs de cluster pour envoyer des messages à partir d'une seule file d'attente de transmission, SYSTEM.CLUSTER.TRANSMIT.QUEUE. La configuration par défaut peut être modifiée en modifiant l'attribut du gestionnaire de files d'attente, **DefClusterXmitQueueType**. La valeur par défaut de l'attribut est SCTQ. Vous pouvez la remplacer par CHANNEL. Si vous définissez l'attribut **DefClusterXmitQueueType** sur CANAL, chaque canal émetteur de cluster utilise par défaut une file d'attente de transmission de cluster spécifique, SYSTEM.CLUSTER.TRANSMIT.ChannelName.

Vous pouvez aussi associer l'attribut de file d'attente de transmission ClusterChannelName à un canal émetteur de cluster manuellement. Les messages destinés au gestionnaire de files d'attente

connecté par le canal émetteur de cluster sont stockés dans la file d'attente de transmission qui identifie le canal émetteur de cluster. Ils ne sont pas stockés dans la file d'attente de transmission du cluster. Si vous associez l'attribut `ClusterChannelName` à des blancs, le canal utilise la file d'attente de transmission du cluster par défaut lorsqu'il redémarre. La file d'attente par défaut est `SYSTEM.CLUSTER.TRANSMIT.ChannelName` ou `SYSTEM.CLUSTER.TRANSMIT.QUEUE`, en fonction de la valeur de l'attribut `DefClusterXmitQueueType` du gestionnaire de files d'attente.

En spécifiant des astérisques, "\*", dans **ClusterChannelName**, vous pouvez associer une file d'attente de transmission à un ensemble de canaux émetteurs de cluster. Ces astérisques peuvent se trouver au début, à la fin ou à plusieurs endroits dans la chaîne de nom de canal. **ClusterChannelName** est limité à une longueur de 20 caractères : `MQ_CHANNEL_NAME_LENGTH`.

### **ClusterName (chaîne de caractères de 48 octets) sur IBM i**

Nom du cluster auquel appartient la file d'attente.

Tableau 763. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X		X	X	

Il s'agit du nom du cluster auquel appartient la file d'attente. Si la file d'attente appartient à plusieurs clusters, `ClusterNameList` spécifie le nom d'un objet liste de noms qui identifie les clusters et `ClusterName` est vide. Au moins l'un des éléments `ClusterName` et `ClusterNameList` doit être vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur `CACLN` avec l'appel `MQINQ`. La longueur de cet attribut est indiquée par `LNCLUN`.

### **ClusterNameList (chaîne de caractères de 48 octets) sur IBM i**

Nom de l'objet liste de noms contenant les noms des clusters auxquels appartient la file d'attente.

Tableau 764. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X		X	X	

Il s'agit du nom d'un objet liste de noms qui contient les noms des clusters auxquels appartient cette file d'attente. Si la file d'attente appartient à un seul cluster, l'objet liste de noms ne contient qu'un seul nom. Vous pouvez également utiliser `ClusterName` pour spécifier le nom du cluster, auquel cas `ClusterNameList` est vide. Au moins l'un des éléments `ClusterName` et `ClusterNameList` doit être vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur `CACNLN` avec l'appel `MQINQ`. La longueur de cet attribut est indiquée par `LNNLN`.

### **CreationDate (chaîne de caractères de 12 octets) sur IBM i**

Date de création de la file d'attente.

Tableau 765. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X				

Il s'agit de la date de création de la file d'attente. Le format de la date est YYYY-MM-DD, complété par deux blancs de fin pour obtenir une longueur de 12 octets (par exemple, 1992-09-23-- , où -- représente deux caractères blancs).

- Sous IBM i, la date de création d'une file d'attente peut être différente de celle de l'entité de système d'exploitation sous-jacente (fichier ou espace utilisateur) qui représente la file d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CACRTD avec l'appel MQINQ. La longueur de cet attribut est donnée par LNCRTD.

### **IBM i** **CreationTime (chaîne de caractères de 8 octets) sur IBM i**

Heure de création de la file d'attente.

Tableau 766. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X				

Il s'agit de l'heure à laquelle la file d'attente a été créée. Le format de l'heure est HH.MM.SS à l'aide de l'horloge au format 24 heures, avec un zéro non significatif si l'heure est inférieure à 10 (par exemple, 09.10.20). L'heure est l'heure locale.

- Sous IBM i, l'heure de création d'une file d'attente peut être différente de celle de l'entité de système d'exploitation sous-jacente (fichier ou espace utilisateur) qui représente la file d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CACRTT avec l'appel MQINQ. La longueur de cet attribut est donnée par LNCRTT.

### **IBM i** **CurrentQDepth (entier signé à 10 chiffres) sur IBM i**

Longueur en cours de la file d'attente.

Tableau 767. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X				

Il s'agit du nombre de messages stockés actuellement dans la file d'attente. Elle est incrémentée lors d'un appel MQPUT et lors de l'annulation d'un appel MQGET. Il est décrémenté lors d'un appel MQGET non navigant et lors de l'annulation d'un appel MQPUT. En conséquence, le nombre inclut les messages qui ont été placés dans la file d'attente dans une unité de travail, mais qui n'ont pas encore été validés, même s'ils ne peuvent pas être extraits par l'appel MQGET. De même, il exclut les messages qui ont été extraits dans une unité de travail à l'aide de l'appel MQGET, mais qui n'ont pas encore été validés.

Ce nombre inclut également les messages dont l'heure d'expiration a été dépassée mais qui n'ont pas encore été supprimés, bien que ces messages ne puissent pas être extraits. Voir la zone *MDEXP* décrite dans «MQMD (descripteur de message) sous IBM i», à la page 1162.

Le traitement de l'unité de travail et la segmentation des messages peuvent tous deux entraîner un dépassement de la valeur de *CurrentQDepth* *MaxQDepth*. Toutefois, cela n'affecte pas la possibilité d'extraction des messages- tous les messages de la file d'attente peuvent être extraits à l'aide de l'appel MQGET de manière normale.

La valeur de cet attribut fluctue au fur et à mesure que le gestionnaire de files d'attente fonctionne.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IACDEP avec l'appel MQINQ.

## ▶ IBM i **DefBind (entier signé à 10 chiffres) sur IBM i**

Liaison par défaut.

Tableau 768. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X		X	X	X

Cet attribut est la liaison par défaut utilisée lorsque OOBNDQ est indiqué dans l'appel MQOPEN et que la file d'attente est une file d'attente de cluster. DefBind peut avoir l'une des valeurs suivantes:

### **BNDOPN**

Liaison corrigée par l'appel MQOPEN.

### **BNDNOT**

Liaison non fixe.

### **BNDGRP**

La liaison n'est pas fixée par l'appel MQOPEN, mais elle est fixée sur MQPUT pour tous les messages d'un groupe logique.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IADBND avec l'appel MQINQ.

## ▶ IBM i **DefinitionType (entier signé à 10 chiffres) sur IBM i**

Type de définition de file d'attente.

Tableau 769. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Indique comment la file d'attente a été définie. La valeur est l'une des suivantes :

### **RPDR**

File d'attente permanente prédéfinie.

La file d'attente est une file d'attente permanente créée par l'administrateur système ; seul l'administrateur système peut la supprimer.

Les files d'attente prédéfinies sont créées à l'aide de la commande DEFINE MQSC et peuvent être supprimées uniquement à l'aide de la commande DELETE MQSC. Les files d'attente prédéfinies ne peuvent pas être créées à partir de files d'attente modèles.

Les commandes peuvent être émises par un opérateur ou par un utilisateur autorisé à envoyer un message de commande à la file d'attente d'entrée de commande (voir l'attribut **CommandInputQName** décrit dans «Attributs du gestionnaire de files d'attente sous IBM i», à la page 1469).

### **QDPERM**

File d'attente permanente définie de manière dynamique.

La file d'attente est une file d'attente permanente créée par une application émettant un appel MQOPEN avec le nom d'une file d'attente modèle indiquée dans le descripteur d'objet MQOD. La définition de file d'attente modèle avait la valeur QDPERM pour l'attribut **DefinitionType**.

Ce type de file d'attente peut être supprimé à l'aide de l'appel MQCLOSE. Pour plus d'informations, voir «MQCLOSE (Fermer l'objet) sur IBM i», à la page 1330.

La valeur de l'attribut **QSGDisp** pour une file d'attente dynamique permanente est QSGDQM.

## QDTEMP

File d'attente temporaire définie de manière dynamique.

La file d'attente est une file d'attente temporaire créée par une application qui émet un appel MQOPEN avec le nom d'une file d'attente modèle indiquée dans le descripteur d'objet MQOD. La définition de file d'attente modèle avait la valeur QDTEMP pour l'attribut **DefinitionType**.

Ce type de file d'attente est supprimé automatiquement par l'appel MQCLOSE lorsqu'il est fermé par l'application qui l'a créé.

La valeur de l'attribut **QSGDisp** pour une file d'attente dynamique temporaire est QSGDQM.

## QDSHAR

File d'attente partagée définie de manière dynamique.

La file d'attente est une file d'attente permanente partagée qui a été créée par une application émettant un appel MQOPEN avec le nom d'une file d'attente modèle indiquée dans le descripteur d'objet MQOD. La définition de file d'attente modèle avait la valeur QDSHAR pour l'attribut **DefinitionType**.

Ce type de file d'attente peut être supprimé à l'aide de l'appel MQCLOSE. Pour plus d'informations, voir «MQCLOSE (Fermer l'objet) sur IBM i», à la page 1330.

La valeur de l'attribut **QSGDisp** pour une file d'attente dynamique partagée est QSGDSH.

Cet attribut dans une définition de file d'attente modèle n'indique pas comment la file d'attente modèle a été définie, car les files d'attente modèles sont toujours prédéfinies. A la place, la valeur de cet attribut dans la file d'attente modèle est utilisée pour déterminer le *DefinitionType* de chacune des files d'attente dynamiques créées à partir de la définition de file d'attente modèle à l'aide de l'appel MQOPEN.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IADEFI avec l'appel MQINQ.



### **DefInputOpenOption (entier signé à 10 chiffres) sur IBM i**

Option d'ouverture d'entrée par défaut.

Tableau 770. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Il s'agit de la méthode par défaut d'ouverture de la file d'attente pour la saisie. Elle s'applique si l'option OOINPQ est indiquée dans l'appel MQOPEN lors de l'ouverture de la file d'attente. Cette valeur peut être l'une des suivantes:

### OOINPX

Ouvrez la file d'attente pour obtenir les messages avec un accès exclusif.

La file d'attente est ouverte pour être utilisée avec les appels MQGET suivants. L'appel échoue avec le code anomalie RC2042 si la file d'attente est actuellement ouverte par cette application ou par une autre pour l'entrée de tout type (OOINPS ou OOINPX).

### OOINPS

Ouvrez la file d'attente pour obtenir les messages avec accès partagé.

La file d'attente est ouverte pour être utilisée avec les appels MQGET suivants. L'appel peut aboutir si la file d'attente est actuellement ouverte par cette application ou par une autre application avec OOINPS, mais échoue avec le code anomalie RC2042 si la file d'attente est actuellement ouverte avec OOINPX.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IADINP avec l'appel MQINQ.

## IBM i **DefPersistence (entier signé à 10 chiffres) sur IBM i**

Persistence de message par défaut.

Tableau 771. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X	X	X	X

Il s'agit de la persistance par défaut des messages dans la file d'attente. Elle s'applique si PEQDEF est spécifié dans le descripteur de message lors de l'insertion du message.

S'il existe plusieurs définitions dans le chemin de résolution de nom de file d'attente, la persistance par défaut est extraite de la valeur de cet attribut dans la *première* définition du chemin au moment de l'appel MQPUT ou MQPUT1 . Possibilités :

- Une file d'attente alias
- Une file d'attente locale
- Définition locale d'une file d'attente éloignée
- Un alias de gestionnaire de files d'attente
- Une file d'attente de transmission (par exemple, la file d'attente *DefXmitQName* )

Cette valeur peut être l'une des suivantes:

### **PEPER**

Le message est persistant.

Cela signifie que le message survit aux échecs du système et aux redémarrages du gestionnaire de files d'attente. Les messages persistants ne peuvent pas être placés sur:

- Files d'attente dynamiques temporaires
- Files d'attente partagées

Les messages persistants peuvent être placés dans des files d'attente dynamiques permanentes et dans des files d'attente prédéfinies.

### **PENPER**

Le message n'est pas persistant.

Cela signifie que le message ne survit normalement pas aux défaillances du système ou aux redémarrages du gestionnaire de files d'attente. Cela s'applique même si une copie intacte du message est trouvée sur la mémoire secondaire lors du redémarrage du gestionnaire de files d'attente.

Dans le cas spécial des files d'attente partagées, les messages non persistants *survivent* aux redémarrages des gestionnaires de files d'attente dans le groupe de partage de files d'attente, mais ne survivent pas aux échecs de l'unité de couplage utilisée pour stocker les messages dans les files d'attente partagées.

Les messages persistants et non persistants peuvent exister dans la même file d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IADPER avec l'appel MQINQ.

## IBM i **DefPriority (entier signé à 10 chiffres) sur IBM i**

Priorité de message par défaut.

Tableau 772. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X	X	X	X

Il s'agit de la priorité par défaut pour les messages de la file d'attente. Ceci s'applique si PRQDEF est indiqué dans le descripteur de message lorsque le message est placé dans la file d'attente.

S'il existe plusieurs définitions dans le chemin de résolution de nom de file d'attente, la priorité par défaut du message est extraite de la valeur de cet attribut dans la *première* définition du chemin au moment de l'opération d'insertion. Possibilités :

- Une file d'attente alias
- Une file d'attente locale
- Définition locale d'une file d'attente éloignée
- Un alias de gestionnaire de files d'attente
- Une file d'attente de transmission (par exemple, la file d'attente *DefXmitQName* )

La manière dont un message est placé dans une file d'attente dépend de la valeur de l'attribut **MsgDeliverySequence** de la file d'attente:

- Si l'attribut **MsgDeliverySequence** est MSPRIO, la position logique à laquelle un message est placé dans la file d'attente dépend de la valeur de la zone *MDPRI* dans le descripteur de message.
- Si l'attribut **MsgDeliverySequence** est MSFIFO, les messages sont placés dans la file d'attente comme s'ils avaient une priorité égale à *DefPriority* de la file d'attente résolue, quelle que soit la valeur de la zone *MDPRI* dans le descripteur de message. Toutefois, la zone *MDPRI* conserve la valeur spécifiée par l'application qui a inséré le message. Pour plus d'informations, voir l'attribut **MsgDeliverySequence** décrit dans «Attributs des files d'attente», à la page 1435 .

Les priorités sont comprises entre zéro (la plus faible) et *MaxPriority* (la plus élevée) ; voir l'attribut **MaxPriority** décrit dans «Attributs du gestionnaire de files d'attente sous IBM i», à la page 1469.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IADPRI avec l'appel MQINQ.

### **DefReadAhead (entier signé à 10 chiffres) sur IBM i**

Indique le comportement de lecture anticipée par défaut pour les messages non persistants distribués au client.

Tableau 773. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X	X		

DefReadAhead peut être défini sur l'une des valeurs suivantes:

#### **RAHNO**

Les messages non persistants ne sont pas envoyés au client avant qu'une application ne les demande. Un maximum d'un message non persistant peut être perdu si le client se termine de manière anormale.

#### **RAHYES**

Les messages non persistants sont envoyés au client avant qu'une application ne les demande. Les messages non persistants peuvent être perdus si le client se termine de manière anormale ou si le client ne consomme pas tous les messages qu'il envoie.

#### **RAHDIS**

La lecture anticipée des messages non persistants n'est pas activée pour cette file d'attente. Les messages ne sont pas envoyés à l'avance au client, que la lecture anticipée soit demandée ou non par l'application client.



Pour déterminer la valeur de cet attribut, utilisez le sélecteur IADRAH avec l'appel MQINQ.

### **IBM i** *DefPResp (entier signé à 10 chiffres) sur IBM i*

L'attribut de type de réponse d'insertion par défaut (DEFRESP) définit la valeur utilisée par les applications lorsque le type PutResponsedans MQPMO a été défini sur PMRASQ. Cet attribut est valide pour tous les types de file d'attente.

<i>Tableau 774. Types de file d'attente auxquels cet attribut s'applique</i>				
<b>Locale</b>	<b>Modèle</b>	<b>Alias</b>	<b>Eloigné</b>	<b>Cluster</b>
X	X	X	X	X

Cette valeur peut être l'une des suivantes:

#### **SYNC**

L'opération d'insertion est émise de manière synchrone et renvoie une réponse.

#### **ASYNC**

L'opération d'insertion est émise de manière asynchrone et renvoie un sous-ensemble de zones MQMD.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IADPRT avec l'appel MQINQ.

### **IBM i** *DistLists (entier signé à 10 chiffres) sur IBM i*

Prise en charge de la liste de distribution.

<i>Tableau 775. Types de file d'attente auxquels cet attribut s'applique</i>				
<b>Locale</b>	<b>Modèle</b>	<b>Alias</b>	<b>Eloigné</b>	<b>Cluster</b>
X	X			

Indique si les messages de liste de distribution peuvent être placés dans la file d'attente. L'attribut est défini par un agent MCA (Message Channel Agent) pour indiquer au gestionnaire de files d'attente local si le gestionnaire de files d'attente à l'autre extrémité du canal prend en charge les listes de distribution. Ce dernier gestionnaire de files d'attente (appelé "gestionnaire de files d'attente partenaire") est celui qui reçoit ensuite le message, après qu'il a été supprimé de la file d'attente de transmission locale par un agent MCA émetteur.

L'attribut est défini par l'agent MCA émetteur chaque fois qu'il établit une connexion à l'agent MCA récepteur sur le gestionnaire de files d'attente partenaire. De cette manière, l'agent MCA émetteur peut amener le gestionnaire de files d'attente local à placer dans la file d'attente de transmission uniquement les messages que le gestionnaire de files d'attente partenaire peut traiter correctement.

Cet attribut est principalement destiné à être utilisé avec les files d'attente de transmission, mais le traitement décrit est effectué quelle que soit l'utilisation définie pour la file d'attente (voir l'attribut **Usage**).

Cette valeur peut être l'une des suivantes:

#### **Utilitaire DLSUPP**

Listes de distribution prises en charge.

Indique que les messages de liste de distribution peuvent être stockés dans la file d'attente et transmis au gestionnaire de files d'attente partenaire sous cette forme. Cela réduit la quantité de traitement requise pour envoyer le message à plusieurs destinations.

#### **DLNSUP**

Les listes de distribution ne sont pas prises en charge.

Cela indique que les messages de liste de distribution ne peuvent pas être stockés dans la file d'attente car le gestionnaire de files d'attente partenaire ne prend pas en charge les listes de distribution. Si une application insère un message de liste de distribution et que ce message doit être placé dans cette file d'attente, le gestionnaire de files d'attente fractionne le message de liste de distribution et place les messages individuels dans la file d'attente à la place. Cela augmente la quantité de traitement requise pour envoyer le message à plusieurs destinations, mais garantit que les messages seront traités correctement par le gestionnaire de files d'attente partenaire.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IADIST avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

### **IBM i HardenGetAnnulation (entier signé à 10 chiffres) sur IBM i**

Indique s'il convient de conserver un nombre d'annulations précis.

Tableau 776. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Pour chaque message, le nombre de fois où le message est extrait par un appel MQGET au sein d'une unité de travail est comptabilisé et cette unité de travail est ensuite annulée. Ce nombre est disponible dans la zone *MDBOC* du descripteur de message une fois l'appel MQGET terminé.

Le nombre d'annulations de message est conservé lorsque le gestionnaire de files d'attente redémarre. Toutefois, pour s'assurer que le nombre est exact, les informations doivent être "renforcées" (enregistrées sur le disque ou sur une autre unité de stockage permanente) chaque fois qu'un message est extrait par un appel MQGET dans une unité de travail pour cette file d'attente. Si cela n'est pas fait et qu'une défaillance du gestionnaire de files d'attente se produit avec l'annulation de l'appel MQGET, le nombre peut ne pas être incrémenté.

Les informations de renforcement pour chaque appel MQGET dans une unité de travail, cependant, imposent un coût de performance et l'attribut **HardenGetBackout** doit être défini sur QABH uniquement si le nombre doit être exact.

- Sous IBM i, le nombre d'annulations de message est toujours renforcé, quelle que soit la valeur de cet attribut.

Les valeurs suivantes sont possibles :

#### **QABH**

Nombre d'annulations mémorisées.

Le renforcement est utilisé pour s'assurer que le nombre d'annulations pour les messages de cette file d'attente est exact.

#### **QABNH**

Il se peut que le nombre d'annulations ne soit pas mémorisé.

Le renforcement n'est pas utilisé pour s'assurer que le nombre d'annulations pour les messages de cette file d'attente est correct. Le nombre peut donc être inférieur à ce qu'il devrait être.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAHGB avec l'appel MQINQ.

### **IBM i InhibitGet (entier signé à 10 chiffres) sur IBM i**

Contrôle si les opérations d'extraction de cette file d'attente sont autorisées.

Tableau 777. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X	X		

Si la file d'attente est une file d'attente alias, les opérations d'extraction doivent être autorisées pour l'alias et la file d'attente de base au moment de l'opération d'extraction, afin que l'appel MQGET aboutisse. La valeur est l'une des suivantes :

#### QAGETI

Les opérations d'extraction sont interdites.

Les appels MQGET échouent avec le code anomalie RC2016. Cela inclut les appels MQGET qui spécifient GMBRWF ou GMBRWN.

**Remarque :** Si un appel MQGET fonctionnant dans une unité de travail aboutit, la modification de la valeur de l'attribut **InhibitGet** après en QAGETI n'empêche pas la validation de l'unité de travail.

#### QAGETA

Les opérations d'extraction sont autorisées.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAIGET avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

### **InhibitPut (entier signé à 10 chiffres) sur IBM i**

Contrôle si les opérations d'insertion pour cette file d'attente sont autorisées.

Tableau 778. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X	X	X	X

S'il existe plusieurs définitions dans le chemin de résolution de nom de file d'attente, les opérations d'insertion doivent être autorisées pour *chaque* définition du chemin (y compris les définitions d'alias de gestionnaire de files d'attente) au moment de l'opération d'insertion, afin que l'appel MQPUT ou MQPUT1 aboutisse. Cette valeur peut être l'une des suivantes:

#### QAPUTI

Les opérations d'insertion sont interdites.

Les appels MQPUT et MQPUT1 échouent avec le code anomalie RC2051.

**Remarque :** Si un appel MQPUT fonctionnant dans une unité de travail aboutit, la modification ultérieure de la valeur de l'attribut **InhibitPut** en QAPUTI n'empêche pas la validation de l'unité de travail.

#### QAPUTA

Les opérations d'insertion sont autorisées.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAIPUT avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

### **InitiationQName (chaîne de caractères de 48 octets) sous IBM i**

Nom de la file d'attente d'initialisation.

Tableau 779. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Il s'agit du nom d'une file d'attente définie sur le gestionnaire de files d'attente local ; la file d'attente doit être de type QTLOC. Le gestionnaire de files d'attente envoie un message de déclenchement à la file d'attente d'initialisation lorsque le démarrage de l'application est requis suite à l'arrivée d'un message dans la file d'attente à laquelle appartient cet attribut. La file d'attente d'initialisation doit être surveillée par une application de moniteur de déclenchement qui démarrera l'application appropriée après réception du message de déclenchement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAINIQ avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNQN.

### **MaxMsgLongueur (entier signé à 10 chiffres) sur IBM i**

Longueur maximale du message en octets.

Tableau 780. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Il s'agit d'une limite supérieure pour la longueur du message *physique* le plus long pouvant être placé dans la file d'attente. Toutefois, comme l'attribut de file d'attente **MaxMsgLength** peut être défini indépendamment de l'attribut de gestionnaire de files d'attente **MaxMsgLength**, la limite supérieure réelle de la longueur du message physique le plus long pouvant être placé dans la file d'attente est la plus faible de ces deux valeurs.

Si le gestionnaire de files d'attente prend en charge la segmentation, il est possible pour une application d'insérer un message *logique* plus long que le moindre des deux attributs **MaxMsgLength**, mais uniquement si l'application spécifie l'indicateur MFSEGA dans MQMD. Si cet indicateur est spécifié, la limite supérieure de la longueur d'un message logique est 999 999 999 999 octets, mais en général, les contraintes de ressources imposées par le système d'exploitation ou par l'environnement dans lequel l'application s'exécute, entraînent une limite inférieure.

Une tentative de placement dans la file d'attente d'un message trop long échoue avec le code anomalie suivant:

- RC2030 si le message est trop grand pour la file d'attente
- RC2031 si le message est trop grand pour le gestionnaire de files d'attente, mais pas trop grand pour la file d'attente

La limite inférieure de l'attribut **MaxMsgLength** est zéro. La limite supérieure est déterminée par l'environnement:

- Sous IBM i, la longueur maximale des messages est de 100 Mo (104 857 600 octets).

Pour plus d'informations, voir le paramètre **BUFLEN** décrit dans «MQPUT (insertion de message) sur IBM i», à la page 1398.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAMLEN avec l'appel MQINQ.

### **MaxQDepth (entier signé à 10 chiffres) sur IBM i**

Longueur maximale de la file d'attente.

Tableau 781. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Il s'agit de la limite supérieure définie pour le nombre de messages physiques pouvant exister simultanément dans la file d'attente. Une tentative d'insertion d'un message dans une file d'attente qui contient déjà des messages *MaxQDepth* échoue avec le code anomalie RC2053.

Le traitement des unités de travail et la segmentation des messages peuvent entraîner un nombre réel de messages physiques dans la file d'attente supérieur à *MaxQDepth*. Toutefois, cela n'affecte pas la possibilité d'extraction des messages- tous les messages de la file d'attente peuvent être extraits à l'aide de l'appel MQGET de manière normale.

La valeur de cet attribut est supérieure ou égale à zéro. La limite supérieure est déterminée par l'environnement.

**Remarque :** Il est possible que l'espace de stockage disponible dans la file d'attente soit épuisé même s'il y a moins de messages *MaxQDepth* dans la file d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAMDEP avec l'appel MQINQ.

### **MediaLog (entier signé à 10 chiffres) sur IBM i**

Identité du domaine réservé aux journaux (ou récepteur de journal sous IBM i) nécessaires à la reprise sur incident lié au support d'une file d'attente particulière.

Tableau 782. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Sur les gestionnaires de files d'attente où la journalisation avec réutilisation automatique des journaux est utilisée, la valeur est renvoyée sous la forme d'une chaîne nulle.

### **MsgDelivery(entier signé à 10 chiffres) sur IBM i**

Séquence de livraison des messages.

Tableau 783. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Cela détermine l'ordre dans lequel les messages sont renvoyés à l'application par l'appel MQGET:

#### **MSFIFO**

Les messages sont renvoyés dans l'ordre FIFO (premier entré, premier sorti).

Cela signifie qu'un appel MQGET renvoie le *premier* message qui répond aux critères de sélection spécifiés sur l'appel, quelle que soit la priorité du message.

#### **MSPRIO**

Les messages sont renvoyés par ordre de priorité.

Cela signifie qu'un appel MQGET renvoie le message *de priorité la plus élevée* qui répond aux critères de sélection spécifiés dans l'appel. Dans chaque niveau de priorité, les messages sont renvoyés dans l'ordre FIFO (premier entré, premier sorti).

Si les attributs appropriés sont modifiés alors que la file d'attente contient des messages, la séquence de distribution est la suivante:

- L'ordre dans lequel les messages sont renvoyés par l'appel MQGET est déterminé par les valeurs des attributs **MsgDeliverySequence** et **DefPriority** en vigueur pour la file d'attente au moment où le message arrive dans la file d'attente:
  - Si *MsgDeliverySequence* est MSFIFO à l'arrivée du message, le message est placé dans la file d'attente comme si sa priorité était *DefPriority*. Cela n'affecte pas la valeur de la zone *MDPRI* dans le descripteur de message du message ; cette zone conserve la valeur qu'elle avait lors de la première insertion du message.
  - Si *MsgDeliverySequence* est MSPRIO à l'arrivée du message, le message est placé dans la file d'attente à l'emplacement approprié à la priorité donnée par la zone *MDPRI* dans le descripteur de message.

Si la valeur de l'attribut **MsgDeliverySequence** est modifiée alors qu'il y a des messages dans la file d'attente, l'ordre des messages dans la file d'attente n'est pas modifié.

Si la valeur de l'attribut **DefPriority** est modifiée alors qu'il y a des messages dans la file d'attente, les messages ne seront pas nécessairement distribués dans l'ordre FIFO, même si l'attribut **MsgDeliverySequence** est défini sur MSFIFO ; ceux qui ont été placés dans la file d'attente à la priorité la plus élevée sont distribués en premier.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAMDS avec l'appel MQINQ.

### **OpenInputNombre (entier signé à 10 chiffres) sur IBM i**

Nombre d'ouvertures pour la saisie.

Tableau 784. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X				

Nombre de descripteurs actuellement valides pour supprimer des messages de la file d'attente à l'aide de l'appel MQGET. Il s'agit du nombre total de descripteurs connus du gestionnaire de files d'attente *local*. Si la file d'attente est une file d'attente partagée, le nombre n'inclut pas les ouvertures pour entrée qui ont été effectuées pour la file d'attente dans d'autres gestionnaires de files d'attente du groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local.

Le nombre inclut les descripteurs dans lesquels une file d'attente alias qui se résout en cette file d'attente a été ouverte en entrée. Le nombre n'inclut pas les descripteurs dans lesquels la file d'attente a été ouverte pour les actions qui n'incluent pas d'entrée (par exemple, une file d'attente ouverte uniquement pour l'exploration).

La valeur de cet attribut fluctue au fur et à mesure que le gestionnaire de files d'attente fonctionne.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAOCI avec l'appel MQINQ.

### **OpenOutputNombre (entier signé à 10 chiffres) sur IBM i**

Nombre d'ouvertures pour la sortie.

Tableau 785. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X				

Il s'agit du nombre de descripteurs actuellement valides pour l'ajout de messages à la file d'attente avec l'appel MQPUT. Il s'agit du nombre total de descripteurs connus du gestionnaire de files d'attente *local* ; il n'inclut pas les ouvertures pour les sorties qui ont été effectuées pour cette file d'attente sur les gestionnaires de files d'attente éloignées. Si la file d'attente est une file d'attente partagée, le nombre n'inclut pas les ouvertures pour sortie qui ont été effectuées pour la file d'attente sur d'autres gestionnaires de files d'attente du groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local.

Le nombre inclut les descripteurs dans lesquels une file d'attente alias qui se résout en cette file d'attente a été ouverte en sortie. Le nombre n'inclut pas les descripteurs où la file d'attente a été ouverte pour les actions qui n'incluent pas de sortie (par exemple, une file d'attente ouverte uniquement pour l'interrogation).

La valeur de cet attribut fluctue au fur et à mesure que le gestionnaire de files d'attente fonctionne.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAOC avec l'appel MQINQ.

### **IBM i** *ProcessName (chaîne de caractères de 48 octets) sur IBM i*

Nom du processus.

Tableau 786. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Il s'agit du nom d'un objet de processus défini sur le gestionnaire de files d'attente local. L'objet processus identifie un programme qui peut traiter la file d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAPRON avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNPRON.

### **IBM i** *QDepthHighEvénement (entier signé à 10 chiffres) sur IBM i*

Contrôle si les événements Longueur élevée de la file d'attente sont générés.

Tableau 787. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Un événement Longueur élevée de file d'attente indique qu'une application a inséré un message dans une file d'attente, ce qui a entraîné un nombre de messages dans la file d'attente supérieur ou égal au seuil maximal de longueur de file d'attente (voir l'attribut **QDepthHighLimit**).

**Remarque :** La valeur de cet attribut peut changer dynamiquement.

L'événement QDepthHigh peut avoir l'une des deux valeurs suivantes:

#### **EVRDIS**

Génération de rapports d'événements désactivée.

#### **EVRENA**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAQDHE avec l'appel MQINQ.

### **IBM i** *QDepthHighLimite (entier signé à 10 chiffres) sur IBM i*

Limite supérieure de la longueur de la file d'attente.

Tableau 788. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Il s'agit du seuil auquel le nombre de lignes de la file d'attente est comparé pour générer un événement Longueur élevée de la file d'attente. Cet événement indique qu'une application a inséré un message dans une file d'attente, ce qui a entraîné un nombre de messages dans la file d'attente supérieur ou égal au seuil maximal de longueur de la file d'attente. Voir l'attribut **QDepthHighEvent** .

La valeur est exprimée en pourcentage de la longueur maximale de la file d'attente (attribut **MaxQDepth** ) et est comprise entre zéro et 100. La valeur par défaut est 80.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAQDHL avec l'appel MQINQ.

### **QDepthLowEvénement (entier signé à 10 chiffres) sur IBM i**

Contrôle si les événements Longueur faible de la file d'attente sont générés.

Tableau 789. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Un événement Longueur basse de la file d'attente indique qu'une application a extrait un message d'une file d'attente, ce qui a entraîné un nombre de messages de la file d'attente inférieur ou égal au seuil bas de longueur de la file d'attente (voir l'attribut **QDepthLowLimit** ).

**Remarque :** La valeur de cet attribut peut changer dynamiquement.

L'événement QDepthLow peut avoir l'une des valeurs suivantes:

#### **EVRDIS**

Génération de rapports d'événements désactivée.

#### **EVRENA**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAQDLE avec l'appel MQINQ.

### **QDepthLowLimite (entier signé à 10 chiffres) sur IBM i**

Limite inférieure de la longueur de la file d'attente.

Tableau 790. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Il s'agit du seuil auquel le nombre de lignes de la file d'attente est comparé pour générer un événement Longueur faible de la file d'attente. Cet événement indique qu'une application a extrait un message d'une file d'attente, ce qui a entraîné un nombre de messages de la file d'attente inférieur ou égal au seuil bas de longueur de la file d'attente. Voir l'attribut **QDepthLowEvent** .

La valeur est exprimée en pourcentage de la longueur maximale de la file d'attente (attribut **MaxQDepth** ) et est comprise entre zéro et 100. La valeur par défaut est 20.



Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAQDLL avec l'appel MQINQ.

**IBM i** **QDepthMaxEvénement (entier signé à 10 chiffres) sur IBM i**  
 Contrôle si les événements File d'attente saturée sont générés.

Tableau 791. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Un événement File d'attente saturée indique qu'une insertion dans une file d'attente a été rejetée car la file d'attente est saturée, c'est-à-dire que la longueur de la file d'attente a déjà atteint sa valeur maximale.

**Remarque :** La valeur de cet attribut peut changer dynamiquement.

Cette valeur peut être l'une des suivantes:

**EVRDIS**

Génération de rapports d'événements désactivée.

**EVRENA**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAQDME avec l'appel MQINQ.

**IBM i** **QDesc (chaîne de caractères de 64 octets) sous IBM i**  
 Description de la file d'attente.

Tableau 792. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X	X	X	X	X

Il s'agit d'une zone qui peut être utilisée pour les commentaires descriptifs. Le contenu de la zone n'est pas significatif pour le gestionnaire de files d'attente, mais ce dernier peut exiger que la zone ne contienne que des caractères pouvant être affichés. Elle ne peut pas contenir de caractères nuls ; si nécessaire, elle est remplie à droite avec des blancs. Dans une installation DBCS, la zone peut contenir des caractères DBCS (avec une longueur de zone maximale de 64 octets).

**Remarque :** Si cette zone contient des caractères qui ne figurent pas dans le jeu de caractères du gestionnaire de files d'attente (tel que défini par l'attribut de gestionnaire de files d'attente **CodedCharSetId**), ces caractères peuvent être convertis de manière incorrecte si cette zone est envoyée à un autre gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAQD avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNQD.

**IBM i** **QName (chaîne de caractères de 48 octets) sous IBM i**  
 Nom de file d'attente.

Tableau 793. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X		X	X	X

Il s'agit du nom d'une file d'attente définie sur le gestionnaire de files d'attente local. Pour plus d'informations sur les noms de file d'attente, voir Règles de dénomination des objets IBM MQ. Toutes les files d'attente définies sur un gestionnaire de files d'attente partagent le même espace de nom de file d'attente. Par conséquent, une file d'attente QTLOC et une file d'attente QTALS ne peuvent pas avoir le même nom.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAQN avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNQN.

### **IBM i** **QServiceInterval (entier signé à 10 chiffres) sur IBM i**

Cible de l'intervalle de service de file d'attente.

Tableau 794. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Il s'agit de l'intervalle de service utilisé à des fins de comparaison pour générer des événements d'intervalle de service élevé et d'intervalle de service OK. Voir l'attribut **QServiceIntervalEvent**.

La valeur est exprimée en millisecondes et est comprise entre zéro et 999 999 999.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAQSI avec l'appel MQINQ.

### **IBM i** **QServiceIntervalEvénement (entier signé à 10 chiffres) sur IBM i**

Contrôle si les événements Intervalle de service élevé ou Intervalle de service OK sont générés.

Tableau 795. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

- Un événement Intervalle de service élevé est généré lorsqu'une vérification indique qu'aucun message n'a été extrait de la file d'attente pendant au moins le temps indiqué par l'attribut **QServiceInterval**.
- Un événement d'intervalle de service OK est généré lorsqu'une vérification indique que des messages ont été extraits de la file d'attente dans le délai indiqué par l'attribut **QServiceInterval**.

**Remarque :** La valeur de cet attribut peut changer dynamiquement.

Cet attribut peut adopter l'une des valeurs suivantes :

#### **QSIEHI**

Événements d'intervalle élevé du service de file d'attente activés.

- Les événements d'intervalle de service de file d'attente élevé sont **activés** et
- Les événements d'intervalle de service de file d'attente OK sont **désactivés**.

#### **QSIEOK**

Événements d'intervalle de service de file d'attente OK activés.

- Les événements d'intervalle de service de file d'attente élevé sont **désactivés** et
- Les événements d'intervalle de service de file d'attente OK sont **activés**.

#### **QSIENO**

Aucun événement d'intervalle de service de file d'attente n'est activé.

- Les événements d'intervalle de service de file d'attente élevé sont **désactivés** et

- Les événements d'intervalle de service de file d'attente OK sont également **désactivés**.

Pour les files d'attente partagées, la valeur de cet attribut est ignorée ; la valeur QSIENO est prise par défaut.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAQSIE avec l'appel MQINQ.

### **QSGDisp (entier signé à 10 chiffres) sous IBM i**

Agencement de groupe de partage de files d'attente.

Tableau 796. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X		X	X	

Indique la disposition de la file d'attente. La valeur est l'une des suivantes :

#### **QSGDQM**

Disposition du gestionnaire de files d'attente.

L'objet possède une disposition de gestionnaire de files d'attente. Cela signifie que la définition d'objet n'est connue que du gestionnaire de files d'attente local ; la définition n'est pas connue des autres gestionnaires de files d'attente du groupe de partage de files d'attente.

Il est possible que chaque gestionnaire de files d'attente du groupe de partage de files d'attente ait un objet ayant le même nom et le même type que l'objet en cours, mais il s'agit d'objets distincts et il n'y a pas de corrélation entre eux. Leurs attributs ne sont pas contraints d'être identiques les uns aux autres.

#### **QSGDCP**

Disposition de l'objet copié.


L'objet est une copie locale d'une définition d'objet maître qui existe dans le référentiel partagé. Chaque gestionnaire de files d'attente du groupe de partage de files d'attente peut disposer de sa propre copie de l'objet. Au départ, toutes les copies ont les mêmes attributs, mais à l'aide des commandes MQSC, chaque copie peut être modifiée de sorte que ses attributs diffèrent de ceux des autres copies. Les attributs des copies sont resynchronisés lorsque la définition principale du référentiel partagé est modifiée.

#### **QSGDSH**

Disposition partagée.

L'objet a une disposition partagée. Cela signifie qu'il existe dans le référentiel partagé une seule instance de l'objet connue de tous les gestionnaires de files d'attente du groupe de partage de files d'attente. Lorsqu'un gestionnaire de files d'attente du groupe accède à l'objet, il accède à l'instance partagée unique de l'objet.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAQSGD avec l'appel MQINQ.

 Cet attribut est pris en charge uniquement sous z/OS.

### **QType (entier signé à 10 chiffres) sur IBM i**

Type de file d'attente.

Tableau 797. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X		X	X	X

Cet attribut comporte l'une des valeurs suivantes :

**QTALS**

Définition de file d'attente alias.

**QTCLUS**

File d'attente de cluster.

**Intervalle QTLOC**

File d'attente locale.

**QTREM**

Définition locale d'une file d'attente éloignée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAQTYP avec l'appel MQINQ.

**IBM i RemoteQMgrNom (chaîne de caractères de 48 octets) sur IBM i**

Nom du gestionnaire de files d'attente éloignées.

Tableau 798. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
			X	

Il s'agit du nom du gestionnaire de files d'attente éloignées sur lequel la file d'attente *RemoteQName* est définie. Si la file d'attente *RemoteQName* a la valeur *QSGDisp* *QSGDCP* ou *QSGDSH*, *RemoteQMgrName* peut être le nom du groupe de partage de files d'attente qui possède *RemoteQName*.

Si une application ouvre la définition locale d'une file d'attente éloignée, *RemoteQMgrName* ne doit pas être vide et ne doit pas être le nom du gestionnaire de files d'attente local. Si *XmitQName* est vide, la file d'attente locale portant le même nom que *RemoteQMgrName* est utilisée comme file d'attente de transmission. S'il n'existe aucune file d'attente portant le nom *RemoteQMgrName*, la file d'attente identifiée par l'attribut de gestionnaire de files d'attente **DefXmitQName** est utilisée.

Si cette définition est utilisée pour un alias de gestionnaire de files d'attente, *RemoteQMgrName* est le nom du gestionnaire de files d'attente dont l'alias est utilisé. Il peut s'agir du nom du gestionnaire de files d'attente local. Sinon, si *XmitQName* est vide lors de l'ouverture, il doit y avoir une file d'attente locale portant le même nom que *RemoteQMgrName* ; cette file d'attente est utilisée comme file d'attente de transmission.

Si cette définition est utilisée pour un alias de réponse, ce nom correspond au nom du gestionnaire de files d'attente qui doit être *MDRM*.

**Remarque :** Aucune validation n'est effectuée sur la valeur spécifiée pour cet attribut lorsque la définition de file d'attente est créée ou modifiée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CARQMN avec l'appel MQINQ. La longueur de cet attribut est donnée par LNQMNM.

**IBM i RemoteQName (chaîne de caractères de 48 octets) sur IBM i**

Nom de la file d'attente éloignée.

Tableau 799. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
			X	

Il s'agit du nom de la file d'attente telle qu'elle est connue sur le gestionnaire de files d'attente éloignées *RemoteQMGrName*.

Si une application ouvre la définition locale d'une file d'attente éloignée, lorsque l'ouverture a lieu, *RemoteQName* ne doit pas être vide.

Si cette définition est utilisée pour une définition d'alias de gestionnaire de files d'attente, lorsque l'ouverture a lieu, la zone *RemoteQName* doit être vide.

Si la définition est utilisée pour un alias de réponse, ce nom correspond au nom de la file d'attente qui doit être *MDRQ*.

**Remarque :** Aucune validation n'est effectuée sur la valeur spécifiée pour cet attribut lorsque la définition de file d'attente est créée ou modifiée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur *CARQN* avec l'appel *MQINQ*. La longueur de cet attribut est indiquée par *LNQN*.

### **RetentionInterval (entier signé à 10 chiffres) sur IBM i**

Intervalle de conservation.

Tableau 800. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Il s'agit de l'heure à laquelle la file d'attente doit être conservée. Une fois ce délai écoulé, la file d'attente peut être supprimée.

La durée est mesurée en heures, à compter de la date et de l'heure de création de la file d'attente. La date de création de la file d'attente est enregistrée dans *CreationDate* et l'heure de création de la file d'attente est enregistrée dans l'attribut **CreationTime**.

Ces informations sont fournies pour permettre à une application de nettoyage ou à l'opérateur d'identifier et de supprimer les files d'attente qui ne sont plus nécessaires.

**Remarque :** Le gestionnaire de files d'attente ne tente jamais de supprimer des files d'attente en fonction de cet attribut, ou d'empêcher la suppression de files d'attente avec un intervalle de conservation qui n'a pas expiré. Il incombe à l'utilisateur d'effectuer les actions requises.

Un intervalle de conservation réaliste doit être utilisé pour empêcher l'accumulation de files d'attente dynamiques permanentes (voir *DefinitionType*). Toutefois, cet attribut peut également être utilisé avec des files d'attente prédéfinies.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur *IARINT* avec l'appel *MQINQ*.

### **Portée (entier signé à 10 chiffres) sur IBM i**

Contrôle si une entrée de cette file d'attente existe également dans un répertoire de cellule.

Tableau 801. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X		X	X	

Un répertoire de cellule est fourni par un service de nom installable. Cette valeur peut être l'une des suivantes:

### **Gestionnaire de files d'attente**

Portée du gestionnaire de files d'attente.

La définition de file d'attente a une portée de type gestionnaire de files d'attente. Cela signifie que la définition de la file d'attente ne s'étend pas au-delà du gestionnaire de files d'attente qui la possède. Pour ouvrir la file d'attente à des fins de sortie à partir d'un autre gestionnaire de files d'attente, le nom du gestionnaire de files d'attente propriétaire doit être spécifié ou l'autre gestionnaire de files d'attente doit avoir une définition locale de la file d'attente.

### **ANNULER**

Portée de la cellule.

La définition de file d'attente a une portée de type cellule. Cela signifie que la définition de file d'attente est également placée dans un répertoire de cellule disponible pour tous les gestionnaires de files d'attente de la cellule. La file d'attente peut être ouverte pour la sortie de n'importe lequel des gestionnaires de files d'attente de la cellule en spécifiant simplement le nom de la file d'attente ; il n'est pas nécessaire de spécifier le nom du gestionnaire de files d'attente propriétaire de la file d'attente. Toutefois, la définition de file d'attente n'est disponible pour aucun gestionnaire de files d'attente de la cellule qui possède également une définition locale d'une file d'attente portant ce nom, car la définition locale est prioritaire.

Un répertoire de cellule est fourni par un service de nom installable tel que LDAP (Lightweight Directory Access Protocol). Notez que IBM MQ ne prend plus en charge le service annuaire DCE (Distributed Computing Environment) qui était auparavant utilisé pour insérer des définitions de file d'attente dans un répertoire DCE (il n'est plus pris en charge non plus).

Les files d'attente modèles et dynamiques ne peuvent pas avoir de portée cellule.

Cette valeur n'est valide que si un service annuaire prenant en charge un répertoire de cellule a été configuré.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IASCOP avec l'appel MQINQ.

La prise en charge de cet attribut est soumise aux restrictions suivantes:

- Sous IBM i, l'attribut est pris en charge, mais seul SCOQM est valide.

### **Partageabilité (entier signé à 10 chiffres) sur IBM i**

Indique si la file d'attente peut être partagée pour l'entrée.

Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Indique si la file d'attente peut être ouverte en entrée plusieurs fois simultanément. Cette valeur peut être l'une des suivantes:

### **QASHR**

La file d'attente est partageable.

Les ouvertures multiples avec l'option OOINPS sont autorisées.

### **QANSHR**

La file d'attente n'est pas partageable.

Un appel MQOPEN avec l'option OOINPS est traité comme OOINPX.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IASHAR avec l'appel MQINQ.

## ▶ IBM i **TriggerControl (entier signé à 10 chiffres) sur IBM i**

Contrôle du déclencheur.

Tableau 803. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Cette option contrôle si les messages de déclenchement sont écrits dans une file d'attente d'initialisation, afin de permettre le démarrage d'une application pour la gestion de la file d'attente. Il s'agit de l'un des éléments suivants:

### **TCOFF**

Messages de déclenchement non requis.

Aucun message de déclenchement ne doit être écrit pour cette file d'attente. La valeur de *TriggerType* n'est pas pertinente dans ce cas.

### **TCON**

Messages de déclenchement requis.

Les messages de déclenchement doivent être écrits pour cette file d'attente lorsque les événements de déclenchement appropriés se produisent.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IATRGC avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

## ▶ IBM i **TriggerData (chaîne de caractères de 64 octets) sur IBM i**

Données de déclenchement.

Tableau 804. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Il s'agit des données de format libre que le gestionnaire de files d'attente insère dans le message de déclenchement lorsqu'un message arrivant dans cette file d'attente entraîne l'écriture d'un message de déclenchement dans la file d'attente d'initialisation.

Le contenu de ces données n'a aucune importance pour le gestionnaire de files d'attente. Il est significatif soit pour l'application du moniteur de déclenchement qui traite la file d'attente d'initialisation, soit pour l'application qui est démarrée par le moniteur de déclenchement.

La chaîne de caractères ne peut pas contenir de valeurs nulles. Il est rempli à droite avec des blancs si nécessaire.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CATRGD avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET. La longueur de cet attribut est indiquée par LNTRGD.

## ▶ IBM i **TriggerDepth (entier signé à 10 chiffres) sur IBM i**

Longueur de déclenchement.

Tableau 805. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Il s'agit du nombre de messages de priorité *TriggerMsgPriority* ou supérieure qui doivent se trouver dans la file d'attente avant qu'un message de déclenchement ne soit écrit. Cela s'applique lorsque *TriggerType* est défini sur TTDPTH. La valeur de *TriggerDepth* est supérieure ou égale à 1. Cet attribut n'est pas utilisé autrement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IATRGD avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

### **IBM i** *TriggerMsgPriority (entier signé à 10 chiffres) sur IBM i*

Priorité des messages de seuil pour les déclencheurs sous IBM MQ for IBM i.

Tableau 806. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Il s'agit de la priorité de message en dessous de laquelle les messages ne contribuent pas à la génération des messages de déclenchement (c'est-à-dire que le gestionnaire de files d'attente ignore ces messages lorsqu'il détermine si un message de déclenchement doit être généré). *TriggerMsgPriority* peut être compris entre zéro (valeur la plus faible) et *MaxPriority* (valeur la plus élevée ; voir «Attributs du gestionnaire de files d'attente sous IBM i», à la page 1469) ; la valeur zéro entraîne la contribution de tous les messages à la génération de messages de déclenchement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IATRGP avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

### **IBM i** *TriggerType (entier signé à 10 chiffres) sur IBM i*

Type de déclencheur.

Tableau 807. Types de file d'attente auxquels cet attribut s'applique

Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Cette option contrôle les conditions dans lesquelles les messages de déclenchement sont écrits suite à l'arrivée de messages dans cette file d'attente. La valeur est l'une des suivantes :

#### **TTNONE**

Aucun message de déclenchement.

Aucun message de déclenchement n'est écrit suite à des messages dans cette file d'attente. Cela a le même effet que de définir *TriggerControl* sur TCOFF.

#### **TTFRST**

Message de déclenchement lorsque la longueur de la file d'attente passe de 0 à 1.

Un message de déclenchement est écrit chaque fois que le nombre de messages de priorité *TriggerMsgPriority* ou supérieure dans la file d'attente passe de 0 à 1.

#### **TTEVRY**

Déclencher un message pour chaque message.



Un message de déclenchement est écrit chaque fois qu'un message de priorité *TriggerMsgPriority* ou supérieure arrive dans la file d'attente.

#### TTDPHTH

Message de déclenchement lorsque le seuil de longueur est dépassé.

Un message de déclenchement est écrit chaque fois que le nombre de messages de priorité *TriggerMsgPriority* ou supérieure dans la file d'attente est égal ou supérieur à *TriggerDepth*. Une fois que le message de déclenchement a été écrit, *TriggerControl* est défini sur TCOFF pour empêcher tout déclenchement supplémentaire jusqu'à ce qu'il soit explicitement activé à nouveau.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IATRGT avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

### **IBM i** *Utilisation (entier signé à 10 chiffres) sur IBM i*

Utilisation de la file d'attente.

Tableau 808. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
X	X			

Indique à quoi sert la file d'attente. La valeur est l'une des suivantes :

#### USNORM

Utilisation normale.

Il s'agit d'une file d'attente que les applications normales utilisent lors de l'insertion et de l'extraction de messages ; la file d'attente n'est pas une file d'attente de transmission.

#### USTRAN

File d'attente de transmission.

Il s'agit d'une file d'attente utilisée pour le stockage des messages destinés aux gestionnaires de files d'attente éloignées. Lorsqu'une application normale envoie un message à une file d'attente éloignée, le gestionnaire de files d'attente local stocke temporairement le message dans la file d'attente de transmission appropriée dans un format spécial. Un agent de canal de transmission lit ensuite le message à partir de la file d'attente de transmission et le transporte vers le gestionnaire de files d'attente éloignées. Pour plus d'informations sur les files d'attente de transmission, voir [Files d'attente de transmission](#).

Seules les applications privilégiées peuvent ouvrir une file d'attente de transmission pour OOOOUT afin d'y placer directement des messages. Seules les applications utilitaires sont normalement censées effectuer cette opération. Vous devez veiller à ce que le format des données de message soit correct (voir «MQXQH (en-tête de file d'attente de transmission) sous IBM i», à la page 1306), sinon des erreurs risquent de se produire lors du processus de transmission. Le contexte n'est pas transmis ou défini sauf si l'une des options de contexte PM\* est spécifiée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAUSAG avec l'appel MQINQ.

### **IBM i** *XmitQName (chaîne de caractères de 48 octets) sur IBM i*

Nom de la file d'attente de transmission.

Tableau 809. Types de file d'attente auxquels cet attribut s'applique				
Locale	Modèle	Alias	Eloigné	Cluster
			X	

Si cet attribut n'est pas à blanc lorsqu'une ouverture se produit, soit pour une file d'attente éloignée, soit pour une définition d'alias de gestionnaire de files d'attente, il indique le nom de la file d'attente de transmission locale à utiliser pour le transfert du message.

Si *XmitQName* est vide, la file d'attente locale portant le même nom que *RemoteQMgrName* est utilisée comme file d'attente de transmission. S'il n'existe aucune file d'attente portant le nom *RemoteQMgrName*, la file d'attente identifiée par l'attribut de gestionnaire de files d'attente **DefXmitQName** est utilisée.

Cet attribut est ignoré si la définition est utilisée en tant qu'alias de gestionnaire de files d'attente et que *RemoteQMgrName* est le nom du gestionnaire de files d'attente local. Il est également ignoré lorsque la définition est utilisée comme définition d'alias de file d'attente de réponses.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAXQN avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNQN.

## Attributs des listes de noms

Cette rubrique récapitule les attributs spécifiques aux listes de noms. Les attributs sont décrits par ordre alphabétique.

**Remarque :** Les noms des attributs affichés sont ceux utilisés avec les appels MQINQ et MQSET.

### Description des attributs

Un objet liste de noms possède les attributs suivants:

#### **AlterationDate (chaîne de caractères de 12 octets)**

Date de la dernière modification de la définition.

Il s'agit de la date de la dernière modification de la définition. Le format de la date est YYYY-MM-DD, complété par deux blancs de fin pour que la longueur soit de 12 octets.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAALTD avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNDATE.

#### **AlterationTime (chaîne de caractères de 8 octets)**

Heure de la dernière modification de la définition.

Il s'agit de l'heure à laquelle la définition a été modifiée pour la dernière fois. Le format de l'heure est HH.MM.SS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAALTT avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNTIME.

#### **NameCount (entier signé à 10 chiffres)**

Nombre de noms dans la liste de noms.

Cette valeur est supérieure ou égale à zéro. La valeur suivante est définie:

##### **NCMXNL**

Nombre maximal de noms dans une liste de noms.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IANAMC avec l'appel MQINQ.

#### **NamelistDesc (chaîne de caractères de 64 octets)**

Description de la liste de noms.

Il s'agit d'une zone qui peut être utilisée pour les commentaires descriptifs ; sa valeur est établie par le processus de définition. Le contenu de la zone n'est pas significatif pour le gestionnaire de files d'attente, mais ce dernier peut exiger que la zone ne contienne que des caractères pouvant être affichés. Elle ne peut pas contenir de caractères nuls ; si nécessaire, elle est remplie à droite avec des blancs. Dans une installation DBCS, cette zone peut contenir des caractères DBCS (avec une longueur de zone maximale de 64 octets).

**Remarque :** Si cette zone contient des caractères qui ne figurent pas dans le jeu de caractères du gestionnaire de files d'attente (tel que défini par l'attribut de gestionnaire de files d'attente **CodedCharSetId**), ces caractères peuvent être convertis de manière incorrecte si cette zone est envoyée à un autre gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CALSTD avec l'appel MQINQ.

La longueur de cet attribut est donnée par LNNLD.

### **NamelistName (chaîne de caractères de 48 octets)**

Nom de la liste de noms.

Il s'agit du nom d'une liste de noms définie sur le gestionnaire de files d'attente local.

Chaque liste de noms a un nom qui est différent des noms des autres listes de noms appartenant au gestionnaire de files d'attente, mais qui peut dupliquer les noms des autres objets de gestionnaire de files d'attente de types différents (par exemple, les files d'attente).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CALSTN avec l'appel MQINQ.

La longueur de cet attribut est indiquée par LNNLN.

### **Noms (chaîne de caractères de 48 octets x NameCount)**

Liste de noms *NameCount* .

Chaque nom correspond au nom d'un objet défini dans le gestionnaire de files d'attente local. Pour plus d'informations sur les noms d'objet, voir [Désignation d'objets IBM MQ](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CANAMS avec l'appel MQINQ.

La longueur de chaque nom dans la liste est donnée par LNOBJN.

## **Attributs des définitions de processus sous IBM i**

Cette rubrique récapitule les attributs spécifiques aux définitions de processus. Les attributs sont décrits par ordre alphabétique.

**Remarque :** Les noms des attributs affichés sont ceux utilisés avec les appels MQINQ et MQSET. Lorsque des commandes MQSC sont utilisées pour définir, modifier ou afficher des attributs, des noms abrégés alternatifs sont utilisés ; voir [Commandes MQSC](#) pour plus de détails.

### **Description des attributs**

Un objet de définition de processus possède les attributs suivants:

#### **AlterationDate (chaîne de caractères de 12 octets)**

Date de la dernière modification de la définition.

Il s'agit de la date de la dernière modification de la définition. Le format de la date est YYYY-MM-DD, complété par deux blancs de fin pour que la longueur soit de 12 octets.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAALTD avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNDATE.

#### **AlterationTime (chaîne de caractères de 8 octets)**

Heure de la dernière modification de la définition.

Il s'agit de l'heure à laquelle la définition a été modifiée pour la dernière fois. Le format de l'heure est HH.MM.SS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAALTT avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNTIME.

#### **ApplId (chaîne de caractères de 256 octets)**

Identificateur d'application.

Il s'agit d'une chaîne de caractères qui identifie l'application à démarrer. Ces informations sont destinées à être utilisées par une application de moniteur de déclenchement qui traite les messages de la file d'attente d'initialisation ; elles sont envoyées à la file d'attente d'initialisation dans le cadre du message de déclenchement.

La signification de *AppLId* est déterminée par l'application trigger-monitor. Le moniteur de déclenchement fourni par IBM MQ requiert que *AppLId* soit le nom d'un programme exécutable.

La chaîne de caractères ne peut pas contenir de valeurs nulles. Il est rempli à droite avec des blancs si nécessaire.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAAPPI avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNPROA.

### **ApplType (entier signé à 10 chiffres)**

Type d'application.

Identifie la nature du programme à démarrer en réponse à la réception d'un message de déclenchement. Ces informations sont destinées à être utilisées par une application de moniteur de déclenchement qui traite les messages de la file d'attente d'initialisation ; elles sont envoyées à la file d'attente d'initialisation dans le cadre du message de déclenchement.

*ApplType* peut avoir n'importe quelle valeur. Vous pouvez utiliser les valeurs suivantes pour les types standard ; les types d'application définis par l'utilisateur sont limités aux valeurs comprises entre ATUFST et ATULST:

#### **àCICS**

Transaction CICS .

#### **AT400**

Application IBM i .

#### **TATUFST**

Valeur la plus faible pour le type d'application défini par l'utilisateur.

#### **ATULST**

Valeur la plus élevée pour le type d'application défini par l'utilisateur.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAAPPT avec l'appel MQINQ.

### **EnvData (chaîne de caractères de 128 octets)**

Données d'environnement.

Il s'agit d'une chaîne de caractères qui contient des informations relatives à l'environnement concernant l'application à démarrer. Ces informations sont destinées à être utilisées par une application de moniteur de déclenchement qui traite les messages de la file d'attente d'initialisation ; elles sont envoyées à la file d'attente d'initialisation dans le cadre du message de déclenchement.

La signification de *EnvData* est déterminée par l'application trigger-monitor. Le moniteur de déclenchement fourni par IBM MQ ajoute *EnvData* à la liste de paramètres transmise à l'application démarrée. La liste de paramètres comprend la structure MQTMC2 , suivie d'un blanc, suivi de *EnvData* avec les blancs de fin supprimés.

La chaîne de caractères ne peut pas contenir de valeurs nulles. Il est rempli à droite avec des blancs si nécessaire.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAENVD avec l'appel MQINQ. La longueur de cet attribut est donnée par LNPROE.

### **ProcessDesc (chaîne de caractères de 64 octets)**

Description du processus.

Il s'agit d'une zone qui peut être utilisée pour les commentaires descriptifs. Le contenu de la zone n'est pas significatif pour le gestionnaire de files d'attente, mais ce dernier peut exiger que la zone ne contienne que des caractères pouvant être affichés. Elle ne peut pas contenir de caractères nuls ;

si nécessaire, elle est remplie à droite avec des blancs. Dans une installation DBCS, la zone peut contenir des caractères DBCS (avec une longueur de zone maximale de 64 octets).

**Remarque :** Si cette zone contient des caractères qui ne figurent pas dans le jeu de caractères du gestionnaire de files d'attente (tel que défini par l'attribut de gestionnaire de files d'attente **CodedCharSetId**), ces caractères peuvent être convertis de manière incorrecte si cette zone est envoyée à un autre gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAPROD avec l'appel MQINQ.

La longueur de cet attribut est indiquée par LNPROD.

### **ProcessName (chaîne de caractères de 48 octets)**

Nom du processus.

Il s'agit du nom d'une définition de processus définie sur le gestionnaire de files d'attente local.

Chaque définition de processus a un nom différent de celui des autres définitions de processus appartenant au gestionnaire de files d'attente. Toutefois, le nom de la définition de processus peut être identique à celui d'autres objets de gestionnaire de files d'attente de types différents (par exemple, des files d'attente).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAPRON avec l'appel MQINQ.

La longueur de cet attribut est indiquée par LNPRON.

### **UserData (chaîne de caractères de 128 octets)**

Données utilisateur.

Il s'agit d'une chaîne de caractères qui contient des informations utilisateur relatives à l'application à démarrer. Ces informations sont destinées à être utilisées par une application de moniteur de déclenchement qui traite les messages dans la file d'attente d'initialisation ou l'application qui est démarrée par le moniteur de déclenchement. Les informations sont envoyées à la file d'attente d'initialisation dans le cadre du message de déclenchement.

La signification de *UserData* est déterminée par l'application trigger-monitor. Le moniteur de déclenchement fourni par IBM MQ transmet *UserData* à l'application démarrée dans le cadre de la liste de paramètres. La liste de paramètres comprend la structure MQTMC2 (contenant *UserData*), suivie d'un blanc, suivie de *EnvData* avec les blancs de fin supprimés.

La chaîne de caractères ne peut pas contenir de valeurs nulles. Il est rempli à droite avec des blancs si nécessaire.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAUSRD avec l'appel MQINQ. La longueur de cet attribut est donnée par LNPROU.

## **IBM i Attributs du gestionnaire de files d'attente sous IBM i**

Récapitulatif des attributs du gestionnaire de files d'attente.

Certains attributs de gestionnaire de files d'attente sont fixes pour des implémentations particulières, tandis que d'autres peuvent être modifiés à l'aide de la commande MQSC ALTER QMGR. Les attributs peuvent également être affichés à l'aide de la commande DISPLAY QMGR. Vous pouvez demander la plupart des attributs de gestionnaire de files d'attente en ouvrant un objet OTQM spécial et en utilisant l'appel MQINQ avec le descripteur renvoyé.

Le tableau suivant récapitule les attributs spécifiques au gestionnaire de files d'attente. Les attributs sont décrits par ordre alphabétique.

**Remarque :** Les noms des attributs affichés dans cette section correspondent aux noms utilisés avec les appels MQINQ et MQSET. Lorsque des commandes MQSC sont utilisées pour définir, modifier ou afficher des attributs, d'autres noms abrégés sont utilisés ; voir [Commandes MQSC](#) pour plus d'informations.

<i>Tableau 810. Attributs du gestionnaire de files d'attente</i>	
<b>Attribut</b>	<b>Description</b>
<u>AlterationDate</u>	Date de la dernière modification de la définition
<u>AlterationTime</u>	Heure de la dernière modification de la définition
<u>AuthorityEvent</u>	Contrôle si des événements d'autorisation (non autorisés) sont générés
<u>BridgeEvent</u>	Contrôle si les événements de pont IMS sont générés
<u>ChannelAutoDef</u>	Contrôle si la définition de canal automatique est autorisée
<u>ChannelAutoDefEvent</u>	Contrôle si des événements de définition automatique de canal sont générés
<u>ChannelAutoDefExit</u>	Nom de l'exit utilisateur pour la définition de canal automatique
<u>ChannelEvent</u>	Contrôle si les événements de canal sont générés
<u>ClusterCacheType</u>	Contrôle si le cache du cluster est de taille fixe ou de taille dynamique
<u>ClusterWorkloadData</u>	Données utilisateur pour l'exit de charge de travail de cluster
<u>ClusterWorkloadExit</u>	Nom de l'exit utilisateur pour la gestion de la charge de travail du cluster
<u>ClusterWorkloadLength</u>	Longueur maximale des données de message transmises à l'exit de charge de travail du cluster
<u>CodedCharSetId</u>	Identificateur de jeu de caractères codés
<u>CommandEvent</u>	Contrôle si les messages d'événement de commande sont mis en file d'attente
<u>CommandInputQName</u>	Nom de la file d'attente d'entrée des commandes
<u>CommandLevel</u>	Niveau de commande
<u>ConfigurationEvent</u>	Événement de configuration
<u>DeadLetterQName</u>	Nom de la file d'attente de rebut
<u>DefClusterXmitQueue</u>	Type de file d'attente de transmission de cluster par défaut
<u>DefXmitQName</u>	Nom de file d'attente de transmission par défaut
<u>DistLists</u>	Prise en charge de la liste de diffusion
<u>InhibitEvent</u>	Contrôle si des événements d'inhibition (Inhibition Get et Inhibition Put) sont générés
<u>LocalEvent</u>	Contrôle si des événements d'erreur locaux sont générés
<u>LoggerEvent</u>	Contrôle si les événements du journal de reprise sont générés
<u>MaxHandles</u>	Nombre maximal de descripteurs
<u>Longueur maximale des messages</u>	Longueur maximale des messages (en octets)
<u>MaxPriority</u>	Priorité maximum
<u>MaxUncommittedMsgs</u>	Nombre maximal de messages non validés dans une unité de travail
<u>PerformanceEvent</u>	Contrôle si les événements liés aux performances sont générés
<u>Plateforme</u>	Plateforme sur laquelle le gestionnaire de files d'attente est en cours d'exécution

<i>Tableau 810. Attributs du gestionnaire de files d'attente (suite)</i>	
<b>Attribut</b>	<b>Description</b>
<u>PubSubMode</u>	Indique si le moteur de publication / abonnement et l'interface de publication / abonnement en file d'attente sont en cours d'exécution
<u>QMgrDesc</u>	Description du gestionnaire de files d'attente
<u>QMgrIdentifier</u>	Identificateur unique généré en interne du gestionnaire de files d'attente
<u>QMgrName</u>	Nom gest. de files
<u>RemoteEvent</u>	Contrôle si des événements d'erreur distants sont générés
<u>RepositoryName</u>	Nom du cluster pour lequel ce gestionnaire de files d'attente fournit des services de référentiel
<u>RepositoryNameList</u>	Nom de l'objet liste de noms contenant les noms des clusters pour lesquels ce gestionnaire de files d'attente fournit des services de référentiel
<u>SSLCRLNameList</u>	Nom de l'objet liste de noms contenant les noms des objets d'informations d'authentification (voir la remarque 1)
<u>SSLEvent</u>	Contrôle si les événements TLS sont générés
<u>SSLKeyRepository</u>	Emplacement du référentiel de clés TLS (voir la remarque 1)
<u>SSLKeyResetNombre</u>	Détermine le nombre d'octets non chiffrés envoyés et reçus dans une conversation TLS avant que la clé de chiffrement ne soit renégociée
<u>StartStopEvent</u>	Contrôle si des événements de démarrage et d'arrêt sont générés
<u>SyncPoint</u>	Prise en charge points sync.
<u>TraceRouteRecording</u>	Contrôle l'enregistrement des informations de routage de trace pour les messages
<u>TreeLifeTime</u>	Durée de vie, en secondes, des rubriques non administratives
<u>TriggerInterval</u>	Intervalle entre les messages de déclenchement
<b>Remarques :</b>	
1. Cet attribut ne peut pas être interrogé à l'aide de l'appel MQINQ et n'est pas décrit dans cette section. Pour plus d'informations sur cet attribut, voir <a href="#">Modification du gestionnaire de files d'attente</a> .	

### **IBM i** ***AlterationDate (chaîne de caractères de 12 octets) sur IBM i***

Date de la dernière modification de la définition.

Il s'agit de la date de la dernière modification de la définition. Le format de la date est YYYY-MM-DD, complété par deux blancs de fin pour que la longueur soit de 12 octets.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAALTD avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNDATE.

### **IBM i** ***AlterationTime (chaîne de caractères de 8 octets) sur IBM i***

Heure de la dernière modification de la définition.

Il s'agit de l'heure à laquelle la définition a été modifiée pour la dernière fois. Le format de l'heure est HH.MM.SS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAALTT avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNTIME.

### **IBM i AuthorityEvent (entier signé à 10 chiffres) sur IBM i**

Contrôle si des événements d'autorisation (non autorisés) sont générés.

L'attribut AuthorityEvent doit être défini sur l'une des valeurs suivantes:

#### **EVRDIS**

Génération de rapports d'événements désactivée.

#### **EVRENA**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAAUTE avec l'appel MQINQ.

### **IBM i BridgeEvent (chaîne de caractères) sur IBM i**

Cet attribut détermine si les messages d'événement de pont IMS sont placés dans SYSTEM.ADMIN.CHANNEL.EVENT EVENT. Il est uniquement pris en charge sous z/OS.

### **IBM i ChannelAutoDef (entier signé à 10 chiffres) sur IBM i**

Contrôle si la définition de canal automatique est autorisée.

Cet attribut contrôle la définition automatique des canaux de type CTCRVR et CTSVCN. Notez que la définition automatique des canaux CTCLSD est toujours activée. Cette valeur peut être l'une des suivantes:

#### **CHADDI**

Définition automatique de canal désactivée.

#### **CHADEN**

Définition automatique de canal activée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IACAD avec l'appel MQINQ.

### **IBM i ChannelAutoDefEvent (entier signé à 10 chiffres) sur IBM i**

Contrôle si des événements de définition automatique de canal sont générés.

Ceci s'applique aux canaux de type CTCRVR, CTSVCN et CTCLSD. Cette valeur peut être l'une des suivantes:

#### **EVRDIS**

Génération de rapports d'événements désactivée.

#### **EVRENA**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance et performances](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IACADE avec l'appel MQINQ.

### **IBM i ChannelAutoDefExit (chaîne de caractères de 20 octets) sur IBM i**

Nom de l'exit utilisateur pour la définition de canal automatique.

Si ce nom n'est pas à blanc et que *ChannelAutoDef* a la valeur CHADEN, l'exit est appelé chaque fois que le gestionnaire de files d'attente est sur le point de créer une définition de canal. Ceci s'applique aux canaux de type CTCRVR, CTSVCN et CTCLSD. L'exit peut alors effectuer l'une des opérations suivantes:

- Autorisez la création de la définition de canal à se poursuivre sans modification.
- Modifiez les attributs de la définition de canal créée.
- Supprimer complètement la création du canal.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CACADX avec l'appel MQINQ. La longueur de cet attribut est donnée par LNEXT.



### **IBM i ChannelEvent (chaîne de caractères) sur IBM i**

Détermine si des messages d'événement de canal sont générés.

Cet attribut détermine si les messages d'événement de canal sont placés dans SYSTEM.ADMIN.CHANNEL.EVENT EVENT et, le cas échéant, type de messages mis en file d'attente (par exemple, 'canal démarré', 'canal arrêté', 'canal non activé'). Avant l'implémentation de cet attribut, le seul moyen d'empêcher la mise en file d'attente des messages d'événement de canal était de supprimer la file d'attente cible.

Cet attribut vous permet également de collecter uniquement les événements de pont IMS (car vous pouvez désormais désactiver les événements de canal, ils ne sont pas placés dans la même file d'attente). Il en est de même pour les événements TLS qui peuvent également être collectés sans qu'il soit nécessaire de collecter également des événements de canal.

Cet attribut vous permet également de collecter des événements significatifs uniquement (par exemple, lorsque des canaux comportent des erreurs, et non lorsqu'ils démarrent et s'arrêtent normalement).

La valeur de l'attribut ChannelEvent peut être l'une des suivantes:

- EVREXP (seuls les événements de canal suivants sont générés: RC2279, RC2283, RC2284, RC2295, RC2296).
- EVRENA (tous les événements de canal sont générés, c'est-à-dire qu'en plus des événements générés par EVREXP, les événements RC2282 et RC2283 sont également générés).
- EVRDIS (aucun événement de canal n'est généré ; il s'agit de la valeur par défaut initiale du gestionnaire de files d'attente).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IACHNE avec l'appel MQINQ.

### **IBM i ClusterCacheType (chaîne de caractères de 32 octets) sur IBM i**

Contrôle si le cache de cluster est de taille fixe ou de taille dynamique.

Il s'agit d'une chaîne de caractères de 32 octets définie par l'utilisateur qui est transmise à l'exit de charge de travail du cluster lorsqu'il est appelé. S'il n'y a pas de données à transmettre à l'exit, la chaîne est vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CACLWD avec l'appel MQINQ.

### **IBM i ClusterWorkloadDonnées (chaîne de caractères de 32 octets) sur IBM i**

Données utilisateur pour l'exit de charge de travail de cluster.

Il s'agit d'une chaîne de caractères de 32 octets définie par l'utilisateur qui est transmise à l'exit de charge de travail du cluster lorsqu'il est appelé. S'il n'y a pas de données à transmettre à l'exit, la chaîne est vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CACLWD avec l'appel MQINQ.

### **IBM i ClusterWorkloadExit (chaîne de caractères de 20 octets) sur IBM i**

Nom de l'exit utilisateur pour la gestion de la charge de travail du cluster.

Si ce nom n'est pas vide, l'exit est appelé chaque fois qu'un message est inséré dans une file d'attente de cluster ou déplacé d'une file d'attente d'émetteur de cluster vers une autre. L'exit peut alors soit accepter l'instance de file d'attente sélectionnée par le gestionnaire de files d'attente comme destination du message, soit sélectionner une autre instance de file d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CACLWX avec l'appel MQINQ. La longueur de cet attribut est donnée par LNEXN.

### **IBM i ClusterWorkloadLongueur (entier signé à 10 chiffres) sur IBM i**

Longueur maximale des données de message transmises à l'exit de charge de travail du cluster.

Longueur maximale des données de message transmises à l'exit de charge de travail du cluster. La longueur réelle des données transmises à l'exit est la valeur minimale suivante:

- Longueur du message.
- Attribut **MaxMsgLength** du gestionnaire de files d'attente.
- Attribut **ClusterWorkloadLength**.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IACLWL avec l'appel MQINQ.

### **IBM i CodedCharSetId (entier signé à 10 chiffres) sur IBM i**

Identificateur de jeu de caractères codés.

Définit le jeu de caractères utilisé par le gestionnaire de files d'attente pour toutes les zones de chaîne de caractères définies dans l'interface MQI, telles que les noms des objets, ainsi que la date et l'heure de création de la file d'attente. Le jeu de caractères doit comporter des caractères mono-octet pour les caractères admis dans les noms d'objet. Elle ne s'applique pas aux données d'application contenues dans le message. La valeur dépend de l'environnement:

- Sous IBM i, la valeur est celle qui est définie dans l'environnement lorsque le gestionnaire de files d'attente est créé pour la première fois.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IACCSI avec l'appel MQINQ.

### **IBM i CommandEvent (entier) sur IBM i**

Contrôle si les messages sont placés dans une file d'attente locale lorsque des commandes sont émises.

Permet de contrôler si les messages sont écrits dans une nouvelle file d'attente d'événements, SYSTEM.ADMIN.COMMAND.EVENT, chaque fois que des commandes sont émises. Cette fonction est utile pour la notification de suivi des commandes et pour le diagnostic des problèmes. Pour vous renseigner sur l'attribut de gestionnaire de files d'attente CommandEvent, utilisez le nouveau sélecteur d'attribut iacev avec l'une des valeurs suivantes:

- EVRENA-Les messages d'événement de commande sont générés et placés dans la file d'attente pour toutes les commandes ayant abouti.
- Les messages d'événement de commande EVND sont générés et placés dans la file d'attente pour toutes les commandes ayant abouti autres que la commande DISPLAY (MQSC) et la commande Inquire (PCF).
- EVRDIS-Les messages d'événement de commande ne sont pas générés ou placés dans la file d'attente (il s'agit de la valeur par défaut initiale du gestionnaire de files d'attente).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CMDEV avec l'appel MQINQ.

### **IBM i CommandInputQName (chaîne de caractères de 48 octets) sur IBM i**

Nom de la file d'attente d'entrée des commandes.

CommandInputQName est le nom de la file d'attente d'entrée de commandes définie sur le gestionnaire de files d'attente local. Il s'agit d'une file d'attente à laquelle les utilisateurs peuvent envoyer des commandes, s'ils y sont autorisés. Le nom de la file d'attente dépend de l'environnement:

- Sous IBM i, le nom de la file d'attente est SYSTEM.ADMIN.COMMAND.QUEUE, et seules les commandes PCF peuvent lui être envoyées. Toutefois, une commande MQSC peut être envoyée à cette file d'attente si la commande MQSC est incluse dans une commande PCF de type CMESC. Pour plus d'informations sur la commande Escape, voir [Escape](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CACMDQ avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNQN.

### **IBM i CommandLevel (entier signé à 10 chiffres) sur IBM i**

Niveau de commande. Indique le niveau des commandes de contrôle du système prises en charge par le gestionnaire de files d'attente.

Le niveau est l'une des valeurs suivantes:

#### **CML800**

Niveau 800 des commandes de contrôle du système.

Cette valeur est renvoyée par les applications suivantes:

- IBM MQ for IBM i
  - version 8.0

#### **CML900**

Niveau 900 des commandes de contrôle du système.

Cette valeur est renvoyée par les applications suivantes:

- IBM MQ for IBM i
  - version 9.0

#### **CML910**

Niveau 910 des commandes de contrôle du système.

Cette valeur est renvoyée par les applications suivantes:

- IBM MQ for IBM i
  - version 9.1

#### **CML920**

Niveau 920 des commandes de contrôle du système.

Cette valeur est renvoyée par les applications suivantes:

- IBM MQ for IBM i
  - Version 9.2

#### **CML930**

Niveau 930 des commandes de contrôle du système.

Cette valeur est renvoyée par les applications suivantes:

- IBM MQ for IBM i
  - Version 9.3

L'ensemble de commandes de contrôle du système qui correspond à une valeur particulière de l'attribut **CommandLevel** varie en fonction de la valeur de l'attribut **Platform**. Ces deux commandes doivent être utilisées pour déterminer les commandes de contrôle du système qui sont prises en charge.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IACMDL avec l'appel MQINQ.

### **ConfigurationEvent sur IBM i**

Contrôle si les événements de configuration sont générés et envoyés à SYSTEM.ADMIN.CONFIG.EVENT .

L'attribut ConfigurationEvent peut prendre l'une des valeurs suivantes:

- EVRENA
- EVRDIS

Si l'attribut ConfigurationEvent est défini sur EVRENA et que certaines commandes sont correctement émises par runmqsc ou PCF, les événements de configuration sont générés et envoyés à SYSTEM.ADMIN.CONFIG.EVENT EVENT. Des événements sont émis pour les commandes suivantes, même si une commande de modification ne modifie pas l'objet impliqué. Les commandes pour lesquelles des événements de configuration sont générés et envoyés sont les suivantes:

- DEFINE/ALTER AUTHINFO
- DEFINIR/MODIFIER LE CANAL

- DEFINE/ALTER NOMELIST
- DEFINE/ALTER PROCESSUS
- DEFINE/ALTER QLOCAL (sauf s'il s'agit d'une file d'attente dynamique temporaire)
- DEFINIR/MODIFIER QMODEL/QALIAS/QREMOTE
- DELETE AUTHINFO
- Supprimer le canal
- Supprimer une liste de noms
- Supprimer un processus
- DELETE QLOCAL (sauf s'il s'agit d'une file d'attente dynamique temporaire)
- SUPPRIMER QMODEL/QALIAS/QREMOTE
- ALTER QMGR (sauf si l'attribut CONFIGEV est désactivé et n'est pas remplacé par activé)
- ACTUALISEZ LE GESTIONNAIRE DE FILES D'ATTENTE
- Un appel MQSET, autre que pour une file d'attente dynamique temporaire.

Les événements ne sont pas générés (s'ils sont activés) dans les cas suivants:

- La commande ou l'appel MQSET a échoué.
- Le gestionnaire de files d'attente ne peut pas placer le message d'événement dans la file d'attente d'événements. La commande doit toujours se terminer avec succès.
- Files d'attente dynamiques temporaires.
- Modifications d'attribut internes effectuées directement ou implicitement (pas par MQSET ou commande) ; cela affecte TRIGGER, CURDEPTH, IPPROCS, OPPOCS, QDPHIEV, QDPLOEV, QDPMAXEV, QSVCI EV.
- Lorsque la file d'attente d'événements de configuration est modifiée, un message d'événement est généré pour cette modification lorsqu'une actualisation est demandée.
- Modifications apportées à la mise en cluster par les commandes REFRESH/RESET CLUSTER et RESUME/SUSPEND QMGR.
- Création ou suppression d'un gestionnaire de files d'attente.

### **DeadLetterQName (chaîne de caractères de 48 octets) sur IBM i**

Nom de la file d'attente de rebut (messages non livrés).

Il s'agit du nom d'une file d'attente définie sur le gestionnaire de files d'attente local. Les messages sont envoyés à cette file d'attente s'ils ne peuvent pas être acheminés vers leur destination correcte.

Par exemple, les messages sont placés dans cette file d'attente lorsque:

- Un message arrive sur un gestionnaire de files d'attente, destiné à une file d'attente qui n'est pas encore définie sur ce gestionnaire de files d'attente
- Un message arrive dans un gestionnaire de files d'attente, mais la file d'attente à laquelle il est destiné ne peut pas le recevoir pour la raison suivante:
  - La file d'attente est saturée
  - Les demandes d'insertion sont interdites
  - Le noeud émetteur n'a pas le droit d'insérer des messages dans la file d'attente

Les applications peuvent également placer des messages dans la file d'attente des messages non livrés.

Les messages de rapport sont traités de la même manière que les messages ordinaires ; si le message de rapport ne peut pas être distribué à sa file d'attente de destination (généralement la file d'attente spécifiée par la zone *MDRQ* dans le descripteur de message du message d'origine), le message de rapport est placé dans la file d'attente de rebut (message non distribué).

**Remarque :** Messages ayant dépassé leur délai d'expiration (voir la zone *MDEXP* décrite dans «MQMD (descripteur de message) sous IBM i», à la page 1162 ) ne sont **pas** transférées à cette file d'attente

lorsqu'elles sont supprimées. Toutefois, un message de rapport d'expiration (ROEXP) est toujours généré et envoyé à la file d'attente *MDRQ*, si l'application émettrice le demande.

Les messages ne sont pas insérés dans la file d'attente de rebut (message non distribué) lorsque l'application qui a émis la demande d'insertion a été informée de manière synchrone du problème avec le code anomalie renvoyé par l'appel MQPUT ou MQPUT1 (par exemple, un message inséré dans une file d'attente locale pour laquelle les demandes d'insertion sont interdites).

Les messages de la file d'attente de messages non livrés ont parfois des données de message d'application préfixées avec une structure MQDLH. Cette structure contient des informations supplémentaires qui indiquent la raison pour laquelle le message a été placé dans la file d'attente des messages non livrés. Pour plus d'informations sur cette structure, voir [«MQDLH \(en-tête de rebut\) sur IBM i»](#), à la page 1115.

Cette file d'attente doit être une file d'attente locale, avec un attribut **Usage** USNORM.

Si une file d'attente de rebut (messages non livrés) n'est pas prise en charge par un gestionnaire de files d'attente ou si aucune file d'attente n'a été définie, le nom est vide. Tous les gestionnaires de files d'attente IBM MQ prennent en charge une file d'attente de rebut (message non distribué), mais elle n'est pas définie par défaut.

Si la file d'attente de rebut (messages non livrés) n'est pas définie ou qu'elle est saturée ou inutilisable pour une autre raison, un message qui lui aurait été transféré par un agent de canal de transmission est conservé dans la file d'attente de transmission.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CADLQ avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNQN.

### ***DefClusterXmitQueueType (entier signé à 10 chiffres)***

L'attribut DefClusterXmitQueue contrôle la file d'attente de transmission sélectionnée par défaut par les canaux émetteurs de cluster à partir de laquelle les messages doivent être envoyés aux canaux récepteurs de cluster.

Les valeurs de **DefClusterXmitQueueType** sont MQCLXQ\_Q\_Q ou MQCLXQ\_CANAL.

#### **MQCLXQ\_SCTQ**

Tous les canaux émetteurs de cluster envoient des messages à partir de SYSTEM.CLUSTER.TRANSMIT.QUEUE. L'ID corréliID de messages placés dans la file d'attente de transmission identifie le canal émetteur de cluster auquel le message est destiné.

SCTQ est défini lorsqu'un gestionnaire de files d'attente est défini.

#### **MQCLXQ\_CHANNEL**

Chaque canal émetteur de cluster envoie des messages à partir d'une file d'attente de transmission différente. Chaque file d'attente de transmission est créée en tant que file d'attente dynamique permanente à partir du modèle de file d'attente SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE.

Si l'attribut de gestionnaire de files d'attente DefClusterXmitQueueType est défini sur CHANNEL, La configuration par défaut est remplacée par une configuration dans laquelle les canaux émetteurs de cluster sont associés à des files d'attente de transmission de cluster individuelles. Les files d'attente de transmission sont des files d'attente dynamiques permanentes créées à partir de la file d'attente modèle SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE. Chacune d'elles est associée à un canal émetteur de cluster. Etant donné qu'un canal émetteur de cluster sert une file d'attente de transmission de cluster, cette dernière contient les messages relatifs à un seul gestionnaire de files d'attente d'un seul cluster. Vous pouvez configurer les clusters afin que chaque gestionnaire de files d'attente d'un cluster contienne une seule file d'attente de cluster. Dans le cas, le trafic de messages entre un gestionnaire de files d'attente et chaque file d'attente de cluster est transféré séparément des files d'attente de messages à d'autres files d'attente.

Pour interroger la valeur, appelez MQINQ ou envoyez une commande Inquire Queue Manager (MQCMD\_INQUIRE\_Q\_MGR) PCF, en définissant le sélecteur MQIA\_DEF\_CLUSTER\_XMIT\_Q\_TYPE. Pour modifier la valeur, envoyez une commande PCF MQCMD\_CHANGE\_Q\_MGR (Modifier un gestionnaire de files d'attente), en définissant le sélecteur MQIA\_DEF\_CLUSTER\_XMIT\_Q\_TYPE.

## Référence associée

[Modifier un gestionnaire de files d'attente](#)

[Consulter les gestionnaires de files d'attente](#)

«MQINQ (interrogation des attributs d'objet) sur IBM i», à la page 1369

L'appel MQINQ renvoie un tableau d'entiers et un ensemble de chaînes de caractères contenant les attributs d'un objet.

### **IBM i** *DefXmitQName (chaîne de caractères de 48 octets) sur IBM i*

Nom de la file d'attente de transmission par défaut.

Il s'agit du nom de la file d'attente de transmission utilisée pour la transmission des messages aux gestionnaires de files d'attente éloignées, s'il n'existe aucune autre indication de la file d'attente de transmission à utiliser.

S'il n'existe pas de file d'attente de transmission par défaut, le nom est entièrement vide. La valeur initiale de cet attribut est vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CADXQN avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNQN.

### **IBM i** *DistLists (entier signé à 10 chiffres) sur IBM i*

Prise en charge de la liste de distribution.

Indique si le gestionnaire de files d'attente local prend en charge les listes de distribution dans les appels MQPUT et MQPUT1. Cette valeur peut être l'une des suivantes:

#### **Utilitaire DLSUPP**

Listes de distribution prises en charge.

#### **DLNSUP**

Les listes de distribution ne sont pas prises en charge.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IADIST avec l'appel MQINQ.

### **IBM i** *InhibitEvent (entier signé à 10 chiffres) sur IBM i*

Contrôle si des événements d'inhibition (Inhibition Get et Inhibition Put) sont générés.

Cette valeur peut être l'une des suivantes:

#### **EVRDIS**

Génération de rapports d'événements désactivée.

#### **EVRENA**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance et performances](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAINHE avec l'appel MQINQ.

### **IBM i** *LocalEvent (entier signé à 10 chiffres) sur IBM i*

Contrôle si des événements d'erreur locaux sont générés.

La valeur est l'une des suivantes :

#### **EVRDIS**

Génération de rapports d'événements désactivée.

#### **EVRENA**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#)

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IALCLE avec l'appel MQINQ.

### IBM i **LoggerEvent (entier signé à 10 chiffres) sur IBM i**

Contrôle si les événements du consignateur de reprise sont générés.

Cette valeur peut être l'une des suivantes:

#### **Activée**

Des événements de consignateur sont générés.

#### **DESACTIVE**

Les événements du consignateur ne sont pas générés. Il s'agit de la valeur par défaut initiale des gestionnaires de files d'attente.

Pour plus d'informations sur les événements, voir [Surveillance et performances](#).

### IBM i **MaxHandles (entier signé à 10 chiffres) sur IBM i**

Nombre maximal de descripteurs.

Il s'agit du nombre maximal de descripteurs ouverts pouvant être utilisés simultanément par une tâche. Chaque appel MQOPEN réussi pour une file d'attente unique (ou pour un objet qui n'est pas une file d'attente) utilise un descripteur. Ce descripteur devient disponible pour être réutilisé lorsque l'objet est fermé. Toutefois, lorsqu'une liste de distribution est ouverte, chaque file d'attente de la liste de distribution se voit attribuer un descripteur distinct, de sorte que l'appel MQOPEN utilise autant de descripteurs que de files d'attente de la liste de distribution. Cela doit être pris en compte lors de la décision d'une valeur appropriée pour *MaxHandles*.

L'appel MQPUT1 effectue un appel MQOPEN dans le cadre de son traitement ; par conséquent, MQPUT1 utilise autant de descripteurs que le ferait MQOPEN, mais les descripteurs ne sont utilisés que pendant la durée de l'appel MQPUT1 lui-même.

La valeur est comprise entre 1 et 999 999 999. Sous IBM i, la valeur par défaut est 256.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAMHND avec l'appel MQINQ.

### IBM i **MaxMsgLongueur (entier signé à 10 chiffres) sur IBM i**

Longueur maximale du message en octets.

Il s'agit de la longueur du message *physique* le plus long pouvant être traité par le gestionnaire de files d'attente. Toutefois, comme l'attribut de gestionnaire de files d'attente **MaxMsgLength** peut être défini indépendamment de l'attribut de file d'attente **MaxMsgLength**, le message physique le plus long pouvant être placé dans une file d'attente est la valeur la moins élevée de ces deux valeurs.

Si le gestionnaire de files d'attente prend en charge la segmentation, il est possible pour une application d'insérer un message *logique* plus long que le moindre des deux attributs **MaxMsgLength**, mais uniquement si l'application spécifie l'indicateur MFSEGA dans MQMD. Si cet indicateur est spécifié, la limite supérieure de la longueur d'un message logique est de 999 999 999 999 octets, mais généralement, les contraintes de ressources imposées par le système d'exploitation ou par l'environnement dans lequel l'application s'exécute, entraîneront une limite inférieure.

La limite inférieure de l'attribut **MaxMsgLength** est de 32 ko (32 768 octets). Sous IBM i, la longueur maximale des messages est de 100 Mo (104 857 600 octets).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAMLEN avec l'appel MQINQ.

### IBM i **MaxPriority (entier signé à 10 chiffres) sur IBM i**

Priorité maximale.

Il s'agit de la priorité de message maximale prise en charge par le gestionnaire de files d'attente. Les priorités sont comprises entre zéro (niveau le plus bas) et *MaxPriority* (niveau le plus élevé).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAMPRI avec l'appel MQINQ.

### IBM i **Messages MaxUncommitted (entier signé à 10 chiffres) sur IBM i**

Nombre maximal de messages non validés dans une unité de travail.

Il s'agit du nombre maximal de messages non validés pouvant exister dans une unité de travail. Le nombre de messages non validés correspond à la somme des éléments suivants depuis le début de l'unité de travail en cours:

- Messages insérés par l'application avec l'option PMSYP
- Messages extraits par l'application avec l'option GMSYP
- Messages de déclenchement et messages de rapport COA générés par le gestionnaire de files d'attente pour les messages insérés avec l'option PMSYP
- Messages de rapport COD générés par le gestionnaire de files d'attente pour les messages extraits avec l'option GMSYP

Les messages suivants ne sont pas comptabilisés comme non validés:

- Messages insérés ou extraits par l'application en dehors d'une unité de travail
- Messages de déclenchement ou messages de rapport COA/COD générés par le gestionnaire de files d'attente suite à l'insertion ou à l'extraction de messages en dehors d'une unité d'oeuvre
- Messages de rapport d'expiration générés par le gestionnaire de files d'attente (même si l'appel à l'origine du message de rapport d'expiration indiqué est GMSYP)
- Messages d'événement générés par le gestionnaire de files d'attente (même si l'appel à l'origine du message d'événement spécifié est PMSYP ou GMSYP)

#### **Remarque :**

1. Les messages de rapport d'exception sont générés par l'agent MCA (Message Channel Agent) ou par l'application et sont donc traités de la même manière que les messages ordinaires insérés ou extraits par l'application.
2. Lorsqu'un message ou un segment est inséré avec l'option PMSYP, le nombre de messages non validés est incrémenté de un, quel que soit le nombre de messages physiques résultant de l'insertion. (Plusieurs messages physiques peuvent être émis si le gestionnaire de files d'attente doit subdiviser le message ou le segment.)
3. Lorsqu'une liste de distribution est placée avec l'option PMSYP, le nombre de messages non validés est incrémenté d'un *pour chaque message physique généré*. Elle peut être aussi petite que 1 ou aussi grande que le nombre de destinations dans la liste de distribution.

La limite inférieure de cet attribut est 1 ; la limite supérieure est 999 999 999.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAMUNC avec l'appel MQINQ.

#### **PerformanceEvent (entier signé à 10 chiffres) sur IBM i**

Contrôle si des événements liés aux performances sont générés.

PerformanceEvent peut avoir l'une des valeurs suivantes:

##### **EVRDIS**

Génération de rapports d'événements désactivée.

##### **EVRENA**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IAPFME avec l'appel MQINQ.

#### **Plateforme (entier signé à 10 chiffres) sur IBM i**

Plateforme sur laquelle le gestionnaire de files d'attente est en cours d'exécution.

Indique le système d'exploitation sur lequel le gestionnaire de files d'attente est en cours d'exécution. La valeur est:

##### **PL400**

IBM i.



## ▶ IBM i **Mode PubSub(entier signé à 10 chiffres) sur IBM i**

Indique si le moteur de publication / abonnement et l'interface de publication / abonnement en file d'attente sont en cours d'exécution, ce qui permet aux applications de publier / abonnement à l'aide de l'interface de programmation d'application et des files d'attente surveillées par l'interface de publication / abonnement en file d'attente.

Cette valeur peut être l'une des suivantes:

### **PSMCP**

Le moteur publication/abonnement est en cours d'exécution. Il est donc possible de publier / s'abonner à l'aide de l'interface de programmation d'application. L'interface de publication / abonnement en file d'attente n'est pas en cours d'exécution. Par conséquent, aucun message inséré dans les files d'attente surveillées par l'interface de publication / abonnement en file d'attente n'est traité. Ce paramètre est utilisé à des fins de compatibilité avec WebSphere Message Broker V6 ou des versions antérieures utilisant ce gestionnaire de files d'attente, car il doit lire les mêmes files d'attente que celles à partir desquelles l'interface de publication / abonnement en file d'attente lit normalement.

### **Fichier PSMDS**

Le moteur pub./abon. et l'interface pub./abon. en file d'attente sont inactifs. Il n'est donc pas possible de publier / s'abonner à l'aide de l'interface de programmation d'application. Les messages de publication / abonnement placés dans les files d'attente surveillées par l'interface de publication / abonnement en file d'attente ne sont pas traités.

### **PSMEN**

Le moteur publication/abonnement et l'interface publication/abonnement sont en cours d'exécution. Il est donc possible de publier / abonner à l'aide de l'interface de programmation d'application et des files d'attente surveillées par l'interface de publication / abonnement en file d'attente. Il s'agit de la valeur par défaut initiale du gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur PSMODE avec l'appel MQINQ.

## ▶ IBM i **QMGrDesc (chaîne de caractères de 64 octets) sur IBM i**

Description du gestionnaire de files d'attente.

Il s'agit d'une zone qui peut être utilisée pour les commentaires descriptifs. Le contenu de la zone n'est pas significatif pour le gestionnaire de files d'attente, mais ce dernier peut exiger que la zone ne contienne que des caractères pouvant être affichés. Elle ne peut pas contenir de caractères nuls ; si nécessaire, elle est remplie à droite avec des blancs. Dans une installation DBCS, cette zone peut contenir des caractères DBCS (avec une longueur de zone maximale de 64 octets).

**Remarque :** Si cette zone contient des caractères qui ne figurent pas dans le jeu de caractères du gestionnaire de files d'attente (tel que défini par l'attribut de gestionnaire de files d'attente **CodedCharSetId**), ces caractères peuvent être convertis de manière incorrecte si cette zone est envoyée à un autre gestionnaire de files d'attente.

Sous IBM i, la valeur par défaut est vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAQMD avec l'appel MQINQ. La longueur de cet attribut est donnée par LNQMD.

## ▶ IBM i **QMGrIdentif (chaîne de caractères de 48 octets) sur IBM i**

Identificateur unique généré en interne du gestionnaire de files d'attente.

Il s'agit d'un nom unique généré en interne pour le gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAQMID avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNQMID.

## ▶ IBM i **QMGrName (chaîne de caractères de 48 octets) sur IBM i**

Nom du gestionnaire de files d'attente

Il s'agit du nom du gestionnaire de files d'attente local, c'est-à-dire du nom du gestionnaire de files d'attente auquel l'application est connectée.

Les 12 premiers caractères du nom sont utilisés pour construire un identificateur de message unique (voir la zone *MDMID* décrite dans «MQMD (descripteur de message) sous IBM i», à la page 1162). Les gestionnaires de files d'attente qui peuvent communiquer entre eux doivent donc avoir des noms qui diffèrent dans les 12 premiers caractères, afin que les identificateurs de message soient uniques dans le réseau du gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CAQMN avec l'appel MQINQ. La longueur de cet attribut est donnée par LNQMN.

### **IBM i RemoteEvent (entier signé à 10 chiffres) sur IBM i**

Contrôle si des événements d'erreur à distance sont générés.

La valeur est l'une des suivantes :

#### **EVRDIS**

Génération de rapports d'événements désactivée.

#### **EVRENA**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IARMTE avec l'appel MQINQ.

### **IBM i RepositoryName (chaîne de caractères de 48 octets) sur IBM i**

Nom du cluster pour lequel ce gestionnaire de files d'attente fournit des services de référentiel.

Il s'agit du nom d'un cluster pour lequel ce gestionnaire de files d'attente fournit un service de gestionnaire de référentiels. Si le gestionnaire de files d'attente fournit ce service pour plusieurs clusters, *RepositoryNameList* spécifie le nom d'un objet liste de noms qui identifie les clusters et *RepositoryName* est vide. Au moins l'un des éléments *RepositoryName* et *RepositoryNameList* doit être vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CARPN avec l'appel MQINQ. La longueur de cet attribut est donnée par LNQMN.

### **IBM i RepositoryNameList (chaîne de caractères de 48 octets) sur IBM i**

Nom de l'objet liste de noms contenant les noms des clusters pour lesquels ce gestionnaire de files d'attente fournit des services de référentiel.

Il s'agit du nom d'un objet liste de noms qui contient les noms des clusters pour lesquels ce gestionnaire de files d'attente fournit un service de gestionnaire de référentiels. Si le gestionnaire de files d'attente fournit ce service pour un seul cluster, l'objet liste de noms ne contient qu'un seul nom. Vous pouvez également utiliser *RepositoryName* pour spécifier le nom du cluster, auquel cas *RepositoryNameList* est vide. Au moins l'un des éléments *RepositoryName* et *RepositoryNameList* doit être vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur CARPNL avec l'appel MQINQ. La longueur de cet attribut est indiquée par LNNLN.

### **IBM i SSLEvent (chaîne de caractères) sous IBM i**

Détermine si des événements TLS sont générés.

La valeur est l'une des suivantes :

- EVRENA (événement MQINQ/PCF/config) ENABLED (MQSC): les événements TLS sont générés (c'est-à-dire que l'événement RC2371 est généré).
- EVRDIS (événement MQINQ/PCF/config) DISABLED (MQSC): Les événements TLS ne sont pas générés. Il s'agit de la valeur par défaut initiale du gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IASSLE avec l'appel MQINQ.

### **SSLKeyResetNombre (entier) sur IBM i**

Détermine le nombre total d'octets non chiffrés qui sont envoyés et reçus dans une conversation TLS, avant que la clé secrète ne soit renégociée. Le nombre d'octets inclut les informations de contrôle envoyées par l'agent MCA.

Cette valeur est utilisée uniquement par les agents MCA de canal TLS qui initient la communication à partir de ce gestionnaire de files d'attente (c'est-à-dire l'agent MCA de canal émetteur dans un appariement de canaux émetteur et récepteur).

Si la valeur de cet attribut est supérieure à 0 et que les pulsations de canal sont activées pour un canal, la clé secrète est également renégociée avant que les données ne soient envoyées ou reçues à la suite d'une pulsation de canal. Nombre d'octets jusqu'à la prochaine renégociation de clé confidentielle après chaque renégociation réussie.

La valeur peut être comprise entre 0 et 999 999 999. La valeur 0 pour cet attribut indique que la clé secrète n'est jamais renégociée. Si vous spécifiez un nombre de réinitialisations de clé confidentielle TLS compris entre 1 octet et 32 Ko, les canaux TLS utiliseront un nombre de réinitialisations de clé confidentielle de 32 Ko. Cela permet d'éviter le coût de traitement d'un trop grand nombre de réinitialisations de clé qui se produiraient pour des valeurs de réinitialisation de clé secrète TLS de petite taille.

Lorsque le serveur SSL est un gestionnaire de files d'attente IBM MQ et que la réinitialisation de clé confidentielle et les pulsations de canal sont activées, la renégociation se produit immédiatement après chaque pulsation de canal.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IASSRC avec l'appel MQINQ.

### **StartStopEvénement (entier signé à 10 chiffres) sur IBM i**

Contrôle si les événements de démarrage et d'arrêt sont générés.

Cet attribut peut adopter l'une des valeurs suivantes :

#### **EVRDIS**

Génération de rapports d'événements désactivée.

#### **EVRENA**

Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir [Surveillance des événements](#).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IASSE avec l'appel MQINQ.

### **SyncPoint (entier signé à 10 chiffres) sur IBM i**

Disponibilité des points de synchronisation.

Indique si le gestionnaire de files d'attente local prend en charge les unités de travail et la synchronisation avec les appels MQGET, MQPUT et MQPUT1 .

#### **SPAVL**

Unités de travail et point de synchronisation disponibles.

#### **SPNAVL**

Unités de travail et point de synchronisation non disponibles.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IASYNC avec l'appel MQINQ.

### **TraceRouteEnregistrement (entier signé à 10 chiffres) sur IBM i**

Permet de contrôler si les informations sur les messages sont enregistrées lorsqu'ils transitent par un gestionnaire de files d'attente.

La valeur est l'une des suivantes :

- RECCD: aucun ajout aux messages de routage de trace n'est autorisé

- RECDQ: les messages sont placés dans une file d'attente nommée fixe
- RECDM: déterminer à l'aide du message (il s'agit du paramètre par défaut initial)

Pour empêcher le message de trace route de rester dans le système, définissez une valeur d'expiration supérieure à zéro et spécifiez l'option de rapport RODISC. Pour éviter que des messages de rapport ou de réponse restent dans le système, définissez l'option de rapport ROPDAE. Pour plus d'informations, voir «Options de rapport et indicateurs de message sous IBM i», à la page 1505.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IATRGI avec l'appel MQINQ.

### **IBM i** *TreeLifeHeure (entier signé à 10 chiffres) sur IBM i*

Durée de vie, en secondes, des rubriques non administratives.

Les rubriques non administratives sont celles créées lorsqu'une application publie dans ou s'abonne en tant que chaîne de rubrique qui n'existe pas en tant que noeud d'administration. Si ce noeud non administratif ne comporte plus d'abonnements actifs, ce paramètre détermine la durée pendant laquelle le gestionnaire de files d'attente attend avant de supprimer ce noeud. Seules les rubriques non administratives utilisées par un abonnement durable sont conservées après recyclage du gestionnaire de files d'attente.

Indiquez une valeur comprise entre 0 et 604 000. La valeur 0 indique que les rubriques non administratives ne sont pas supprimées par le gestionnaire de files d'attente. La valeur par défaut initiale du gestionnaire de files d'attente est 1800.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IATRLFT avec l'appel MQINQ.

### **IBM i** *TriggerInterval (entier signé à 10 chiffres) sur IBM i*

Intervalle entre les messages de déclenchement.

Il s'agit d'un intervalle de temps (en millisecondes) utilisé pour limiter le nombre de messages de déclenchement. Cela n'est pertinent que lorsque le *TriggerType* est TFRST. Dans ce cas, les messages de déclenchement sont normalement générés uniquement lorsqu'un message approprié arrive dans la file d'attente et que la file d'attente était précédemment vide. Dans certaines circonstances, cependant, un message de déclenchement supplémentaire peut être généré avec le déclenchement TFRST même si la file d'attente n'était pas vide. Ces messages de déclenchement supplémentaires ne sont pas générés plus souvent que toutes les *TriggerInterval* millisecondes.

Pour plus d'informations sur le déclenchement, voir [Déclenchement de canaux](#).

La valeur est comprise entre zéro et 999 999 999. La valeur par défaut est 999 999 999.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur IATRGI avec l'appel MQINQ.

## Applications

Ces informations décrivent les exemples de programmes fournis avec IBM MQ for IBM i for RPG. Apprenez également à générer des applications exécutables à partir des programmes que vous écrivez.

### Génération de votre application

Les publications IBM i expliquent comment générer des applications exécutables à partir des programmes que vous écrivez. Cette rubrique décrit les tâches supplémentaires et les modifications apportées aux tâches standard que vous devez effectuer lors de la génération d'applications IBM MQ for IBM i à exécuter sous IBM i.

En plus de coder les appels MQI dans votre code source, vous devez ajouter les instructions de langage appropriées pour inclure les fichiers de copie IBM MQ for IBM i pour le langage RPG. Vous devez vous familiariser avec le contenu de ces fichiers ; leurs noms et une brève description de leur contenu sont donnés dans le texte suivant.

## IBM i **IBM MQ copier les fichiers sur IBM i**

IBM MQ for IBM i fournit des fichiers de copie pour vous aider à écrire vos applications dans le langage de programmation RPG. Ils peuvent être utilisés avec le compilateur ILE RPG 4 de l'ensemble d'outils de développement WebSphere (5722 WDS).

Les fichiers de copie fournis par IBM MQ for IBM i pour faciliter l'écriture des exits de canal sont décrits dans [Programmes d'exit de canal pour les canaux de messagerie](#).

Les noms des fichiers de copie IBM MQ for IBM i pour RPG comportent le préfixe CMQ. Ils ont un suffixe G ou H. Il existe des fichiers de copie distincts contenant les constantes nommées et un fichier pour chacune des structures. Les fichiers de copie sont répertoriés dans le [«Considérations linguistiques»](#), à la page 1057.

**Remarque :** Pour ILE RPG/400, ils sont fournis en tant que membres de fichier QRPGLSRC dans la bibliothèque QMQM.

Les déclarations de structure ne contiennent pas d'instructions DS . Cela permet à l'application de déclarer une structure de données (ou une structure de données à occurrences multiples) en codant l'instruction DS et en utilisant l'instruction /COPY pour la copier dans le reste de la déclaration:

Pour ILE RPG/400 , l'instruction est la suivante:

```
D*..1.....2.....3.....4.....5.....6.....7
D* Declare an MQMD data structure
D MQMD      DS
D/COPY CMQMDG
```

### **Préparation de l'exécution de vos programmes**

Pour créer une application IBM MQ for IBM i exécutable, vous devez compiler le code source que vous avez écrit.

Pour effectuer cette opération pour ILE RPG/400, vous pouvez utiliser les commandes IBM i classiques, CRTRPGMOD et CRTPGM.

Après avoir créé votre module \*MODULE, vous devez spécifier BNDSRVPGM(QMQM/LIBMQM) dans la commande CRTPGM. Cela inclut les différentes procédures IBM MQ de votre programme.

Assurez-vous que la bibliothèque contenant les fichiers de copie (QMQM) figure dans la liste des bibliothèques lorsque vous effectuez la compilation.

Pour plus d'informations sur les considérations relatives à la programmation, y compris les modes client, voir [«Considérations linguistiques»](#), à la page 1057.

### **Interfaces avec le gestionnaire de points de synchronisation externe IBM i**

IBM MQ for IBM i utilise le contrôle de validation IBM i natif comme coordinateur de point de synchronisation externe.

Pour plus d'informations sur les fonctions de contrôle de validation d' IBM i, voir *IBM i Programming: Backup and Recovery Guide* .

Pour démarrer les fonctions de contrôle de validation IBM i , utilisez la commande système STRCMTCTL. Pour arrêter le contrôle de validation, utilisez la commande système ENDCMTCTL.

**Remarque :** La valeur par défaut de *Portée de la définition de validation* est \*ACTGRP. Il doit être défini comme \*JOB pour IBM MQ for IBM i. Exemple :

```
STRCMTCTL LCKLVL(*ALL) CMTSCOPE(*JOB)
```

Si vous appelez MQPUT, MQPUT1ou MQGET, en spécifiant PMSYP ou GMSYP, après avoir démarré le contrôle de validation, IBM MQ for IBM i s'ajoute en tant que ressource de validation d'API à la définition de validation. Il s'agit généralement du premier appel de ce type dans un travail. Si des ressources

de validation d'API sont enregistrées dans une définition de validation particulière, vous ne pouvez pas mettre fin au contrôle de validation pour cette définition.

IBM MQ for IBM i supprime son enregistrement en tant que ressource de validation d'API lorsque vous vous déconnectez du gestionnaire de files d'attente, à condition qu'il n'y ait pas d'opérations MQI en attente dans l'unité de travail en cours.

Si vous vous déconnectez du gestionnaire de files d'attente alors que des opérations MQPUT, MQPUT1 ou MQGET sont en attente dans l'unité d'oeuvre en cours, IBM MQ for IBM i reste enregistré en tant que ressource de validation d'API afin d'être informé de la prochaine validation ou annulation. Lorsque le point de synchronisation suivant est atteint, IBM MQ valide ou annule les modifications selon les besoins. Il est possible pour une application de se déconnecter et de se reconnecter à un gestionnaire de files d'attente pendant une unité d'oeuvre active et d'effectuer d'autres opérations MQGET et MQPUT dans la même unité d'oeuvre (il s'agit d'une déconnexion en attente).

Si vous tentez d'émettre une commande système ENDCMTCTL pour cette définition de validation, le message CPF8355 est émis, indiquant que les modifications en attente étaient actives. Ce message apparaît également dans l'historique du travail à la fin du travail. Pour éviter cela, vérifiez que vous validez ou annulez toutes les opérations IBM MQ en attente et que vous vous déconnectez du gestionnaire de files d'attente. Par conséquent, l'utilisation des commandes COMMIT ou ROLLBACK avant ENDCMTCTL doit permettre au contrôle de fin de validation de s'exécuter correctement.

Lorsque le contrôle de validation IBM i est utilisé en tant que coordinateur de point de synchronisation externe, les appels MQCMIT, MQBACK et MQBEGIN peuvent ne pas être émis. Les appels à ces fonctions échouent avec le code anomalie RC2012.

Pour valider ou annuler (c'est-à-dire annuler) votre unité de travail, utilisez l'un des langages de programmation prenant en charge le contrôle de validation. Exemple :

- Commandes CL: COMMIT et ROLLBACK
- Fonctions de programmation ILE C: \_Rcommit et \_Rrollback
- RPG/400: COMMIT et ROLBK
- COBOL/400: COMMIT et ROLLBACK

### ***Points de synchronisation dans les applications CICS for IBM i***

IBM MQ for IBM i participe à des unités d'oeuvre avec CICS. Vous pouvez utiliser l'interface MQI dans une application CICS pour insérer et extraire des messages dans l'unité d'oeuvre en cours.

Vous pouvez utiliser la commande EXEC CICS SYNCPOINT pour établir un point de synchronisation qui inclut les opérations IBM MQ for IBM i. Pour annuler toutes les modifications jusqu'au point de synchronisation précédent, vous pouvez utiliser la commande EXEC CICS SYNCPOINT ROLLBACK.

Si vous utilisez l'option MQPUT, MQPUT1 ou MQGET avec l'option PMSYP ou GMSYP définie dans une application CICS, vous ne pouvez pas vous déconnecter de CICS tant que IBM MQ for IBM i n'a pas supprimé son enregistrement en tant que ressource de validation d'API. Par conséquent, vous devez valider ou déconnecter toutes les opérations d'insertion ou d'extraction en attente avant de vous déconnecter du gestionnaire de files d'attente. Cela vous permettra de vous déconnecter de CICS.

### **Exemples de programmes sous IBM i**

Cette rubrique décrit les exemples de programmes fournis avec IBM MQ for IBM i for RPG. Les exemples illustrent des utilisations typiques de l'interface MQI (Message Queue Interface).

Les exemples ne sont pas destinés à démontrer des techniques de programmation générales. Par conséquent, certaines vérifications d'erreur que vous pouvez souhaiter inclure dans un programme de production ont été omises. Toutefois, ces exemples peuvent être utilisés comme base pour vos propres programmes de mise en file d'attente de messages.

Le code source de tous les exemples est fourni avec le produit ; cette source inclut des commentaires qui expliquent les techniques de mise en file d'attente des messages présentées dans les programmes.

Il existe un ensemble d'exemples de programmes ILE:

### 1. Programmes utilisant des appels prototypés à l'interface MQI (appels liés statiques)

La source existe dans QMQMSAMP/QRPGLESRC. Les membres sont nommés AMQ3xxx4, où xxx indique l'exemple de fonction. Des membres de copie existent dans QMQM/QRPGLESRC. Chaque nom de membre possède un suffixe G ou H.

Le Tableau 811, à la page 1487 fournit la liste complète des exemples de programmes fournis avec IBM MQ for IBM i et indique les noms des programmes dans chacun des langages de programmation pris en charge. Notez que leurs noms commencent tous par le préfixe AMQ, le quatrième caractère du nom indiquant le langage de programmation.

<i>Tableau 811. Noms des exemples de programmes</i>	
	<b>RPG (ILE)</b>
Exemples d'insertion	AMQ3PUT4
Parcourir les exemples	AMQ3GBR4
Obtenir des exemples	AMQ3GET4
Exemples de demande	AMQ3REQ4
Echantillons d'écho	AMQ3ECH4
Consulter les exemples	AMQ3INQ4
Définir des exemples	AMQ3SET4
Exemple de moniteur de déclenchement	AMQ3TRG4
Exemple de serveur de déclenchement	AMQ3SRV4

En plus de ces éléments, l'exemple d'option IBM MQ for IBM i inclut un exemple de fichier de données, AMQSDATA, qui peut être utilisé comme entrée pour certains exemples de programmes et d'exemples de programmes CL illustrant les tâches d'administration. Les exemples CL sont décrits dans [Administration d'IBM i](#). Vous pouvez utiliser l'exemple de programme CL pour créer des files d'attente à utiliser avec les exemples de programme décrits dans cette rubrique.

Pour plus d'informations sur l'exécution des exemples de programme, voir [«Préparation et exécution des exemples de programmes sous IBM i»](#), à la page 1488.

### **Fonctions démontrées dans les exemples de programme sous IBM i**

Tableau présentant les techniques mises en évidence par les exemples de programme IBM MQ for IBM i.

Certaines techniques sont utilisées dans plus d'un programme exemple, mais un seul programme est répertorié dans le tableau. Tous les exemples ouvrent et ferment des files d'attente à l'aide des appels MQOPEN et MQCLOSE. Ces techniques ne sont donc pas répertoriées séparément dans le tableau.

<i>Tableau 812. Exemples de programmes illustrant l'utilisation de l'interface MQI</i>	
<b>Technique</b>	<b>RPG (ILE)</b>
Utilisation des appels MQCONN et MQDISC	AMQ3ECH4 ou AMQ3INQ4
Connexion et déconnexion implicites	AMQ3PUT4
Insertion de messages à l'aide de l'appel MQPUT	AMQ3PUT4
Insertion d'un message unique à l'aide de l'appel MQPUT1	AMQ3ECH4 ou AMQ3INQ4
Réponse à un message de demande	AMQ3INQ4

<i>Tableau 812. Exemples de programmes illustrant l'utilisation de l'interface MQI (suite)</i>	
<b>Technique</b>	<b>RPG (ILE)</b>
Obtention de messages (pas d'attente)	AMQ3GBR4
Obtention de messages (attente avec une limite de temps)	AMQ3GET4
Obtention de messages (avec conversion de données)	AMQ3ECH4
Exploration d'une file d'attente	AMQ3GBR4
Utilisation d'une file d'entrée partagée	AMQ3INQ4
Utilisation d'une file d'attente en entrée exclusive	AMQ3REQ4
Utilisation de l'appel MQINQ	AMQ3INQ4
Utilisation de l'appel MQSET	AMQ3SET4
Utilisation d'une file d'attente de réponse	AMQ3REQ4
Demande de messages d'exception	AMQ3REQ4
Acceptation d'un message tronqué	AMQ3GBR4
Utilisation d'un nom de file d'attente résolue	AMQ3GBR4
Traitement du déclencheur	AMQ3SRV4 ou AMQ3TRG4

**Remarque :** Tous les exemples de programme produisent un fichier spoule qui contient les résultats du traitement.

### ***Préparation et exécution des exemples de programmes sous IBM i***

Avant de pouvoir exécuter les exemples de programme IBM MQ for IBM i , vous devez les compiler comme vous le feriez pour d'autres applications IBM MQ for IBM i . Pour ce faire, vous pouvez utiliser les commandes IBM i CRTRPGMOD et CRTPGM.

Lorsque vous créez les programmes AMQ3xxx4 , vous devez spécifier BNDSRVPGM (QMQM/LIBMQM) dans la commande CRTPGM. Cette opération inclut les différentes procédures IBM MQ de votre programme.

Les exemples de programme sont fournis dans la bibliothèque QMQMSAMP en tant que membres de QRPGLSRC. Ils utilisent les fichiers de copie fournis dans la bibliothèque QMQM. Par conséquent, assurez-vous que cette bibliothèque figure dans la liste des bibliothèques lorsque vous les compilez. Le compilateur RPG fournit des messages d'information car les exemples n'utilisent pas un grand nombre de variables déclarées dans les fichiers de copie.

### **Exécution des exemples de programme**

Vous pouvez utiliser vos propres files d'attente lorsque vous exécutez les exemples ou vous pouvez compiler et exécuter AMQSAMP4 pour créer des exemples de files d'attente. La source de ce programme est fournie dans le fichier QCLSRC de la bibliothèque QMQMSAMP. Il peut être compilé à l'aide de la commande CRTCLPGM.

Pour appeler l'un des exemples de programme, utilisez une commande telle que la suivante:

```
CALL PGM(QMQMSAMP/AMQ3PUT4) PARM('Queue_Name','Queue_Manager_Name')
```

Où Queue\_Name et Queue\_Manager\_Name doivent avoir une longueur de 48 caractères, ce que vous obtenez en complétant Queue\_Name et Queue\_Manager\_Name avec le nombre requis de blancs.

Pour les exemples de programme Inquire and Set, les exemples de définition créés par AMQSAMP4 déclenchent les versions C de ces exemples. Si vous souhaitez déclencher les versions RPG, vous devez



modifier les définitions de processus SYSTEM.SAMPLE.ECHOPROCESS et SYSTEM.SAMPLE.INQPROCESS et SYSTEM.SAMPLE.SETPROCESS. Vous pouvez utiliser la commande CHGMQMPRC (décrite dans [Modification du processus MQ \(CHGMQMPRC\)](#)) pour ce faire, ou éditez et exécutez AMQSAMP4 avec la définition alternative.

### **Exemple de programme Put sous IBM i**

L'exemple de programme d'insertion, AMQ3PUT4, insère des messages dans une file d'attente à l'aide de l'appel MQPUT.

Pour démarrer le programme, appelez le programme et indiquez le nom de votre file d'attente cible comme paramètre de programme. Le programme place un ensemble de messages fixes dans la file d'attente ; ces messages sont extraits du bloc de données à la fin du code source du programme. Un exemple de programme d'insertion est AMQ3PUT4 dans la bibliothèque QMQMSAMP.

A l'aide de cet exemple de programme, la commande est la suivante:

```
CALL PGM(QMQMSAMP/AMQ3PUT4) PARM('Queue_Name','Queue_Manager_Name')
```

Où Queue\_Name et Queue\_Manager\_Name doivent avoir une longueur de 48 caractères, ce que vous obtenez en complétant Queue\_Name et Queue\_Manager\_Name avec le nombre requis de blancs.

### **Conception de l'exemple de programme Put**

Le programme utilise l'appel MQOPEN avec l'option OOOOUT pour ouvrir la file d'attente cible afin d'insérer des messages. Les résultats sont générés dans un fichier spoule. S'il ne parvient pas à ouvrir la file d'attente, le programme écrit un message d'erreur contenant le code anomalie renvoyé par l'appel MQOPEN. Pour simplifier le programme, sur cet appel MQI et sur les appels MQI suivants, le programme utilise des valeurs par défaut pour de nombreuses options.

Pour chaque ligne de données contenue dans le code source, le programme lit le texte dans une mémoire tampon et utilise l'appel MQPUT pour créer un message de datagramme contenant le texte de cette ligne. Le programme se poursuit jusqu'à ce qu'il atteigne la fin de l'entrée ou que l'appel MQPUT échoue. Si le programme atteint la fin de l'entrée, il ferme la file d'attente à l'aide de l'appel MQCLOSE.

### **L'exemple de programme Browse sur IBM i**

L'exemple de programme Parcourir, AMQ3GBR4, parcourt les messages d'une file d'attente à l'aide de l'appel MQGET.

Le programme extrait des copies de tous les messages de la file d'attente que vous indiquez lorsque vous appelez le programme ; les messages restent dans la file d'attente. Vous pouvez utiliser la file d'attente fournie SYSTEM.SAMPLE.LOCAL; exécutez d'abord l'exemple de programme Put pour insérer des messages dans la file d'attente. Vous pouvez utiliser la file d'attente SYSTEM.SAMPLE.ALIAS, qui est un nom d'alias pour la même file d'attente locale. Le programme se poursuit jusqu'à ce qu'il atteigne la fin de la file d'attente ou qu'un appel MQI échoue.

Voici un exemple de commande permettant d'appeler le programme RPG:

```
CALL PGM(QMQMSAMP/AMQ3GBR4) PARM('Queue_Name','Queue_Manager_Name')
```

Où Queue\_Name et Queue\_Manager\_Name doivent avoir une longueur de 48 caractères, ce que vous obtenez en complétant Queue\_Name et Queue\_Manager\_Name avec le nombre requis de blancs. Par conséquent, si vous utilisez SYSTEM.SAMPLE.LOCAL comme file d'attente cible, vous aurez besoin de 29 caractères blancs.

### **Conception de l'exemple de programme Parcourir**

Le programme ouvre la file d'attente cible à l'aide de l'appel MQOPEN avec l'option OOBROW. S'il ne parvient pas à ouvrir la file d'attente, le programme écrit un message d'erreur dans son fichier spoule, contenant le code anomalie renvoyé par l'appel MQOPEN.

Pour chaque message de la file d'attente, le programme utilise l'appel MQGET pour copier le message à partir de la file d'attente, puis affiche les données contenues dans le message. L'appel MQGET utilise les options suivantes:

#### **GMBRWN**

Après l'appel MQOPEN, le curseur de navigation est positionné logiquement avant le premier message de la file d'attente. Par conséquent, cette option entraîne le renvoi du *premier* message lorsque l'appel est effectué pour la première fois.

#### **GMNWT**

Le programme n'attend pas s'il n'y a pas de messages dans la file d'attente.

#### **GMATM**

L'appel MQGET spécifie une mémoire tampon de taille fixe. Si un message est plus long que cette mémoire tampon, le programme affiche le message tronqué, ainsi qu'un avertissement indiquant que le message a été tronqué.

Le programme montre comment vous devez effacer les zones *MDMID* et *MDCID* de la structure MQMD après chaque appel MQGET car l'appel définit ces zones sur les valeurs contenues dans le message qu'il extrait. L'effacement de ces zones signifie que les appels MQGET successifs extraient les messages dans l'ordre dans lequel ils sont placés dans la file d'attente.

Le programme se poursuit jusqu'à la fin de la file d'attente ; ici, l'appel MQGET renvoie le code anomalie RC2033 (aucun message disponible) et le programme affiche un message d'avertissement. Si l'appel MQGET échoue, le programme écrit un message d'erreur qui contient le code anomalie dans son fichier spoule.

Le programme ferme ensuite la file d'attente à l'aide de l'appel MQCLOSE.

### ***L'exemple de programme Get sur IBM i***

L'exemple de programme d'extraction, AMQ3GET4, extrait des messages d'une file d'attente à l'aide de l'appel MQGET.

Lorsque le programme est appelé, il supprime les messages de la file d'attente indiquée. Vous pouvez utiliser la file d'attente fournie SYSTEM.SAMPLE.LOCAL; exécutez d'abord l'exemple de programme Put pour insérer des messages dans la file d'attente. Vous pouvez utiliser SYSTEM.SAMPLE.ALIAS, qui est un nom d'alias pour la même file d'attente locale. Le programme se poursuit jusqu'à ce que la file d'attente soit vide ou qu'un appel MQI échoue.

Voici un exemple de commande permettant d'appeler le programme RPG:

```
CALL PGM(QMQMSAMP/AMQ3GET4) PARM('Queue_Name', 'Queue_Manager_Name')
```

où *Queue\_Name* et *Queue\_Manager\_Name* doivent avoir une longueur de 48 caractères, ce que vous obtenez en complétant *Queue\_Name* et *Queue\_Manager\_Name* avec le nombre requis de blancs. Par conséquent, si vous utilisez SYSTEM.SAMPLE.LOCAL comme file d'attente cible, vous aurez besoin de 29 caractères blancs.

### **Conception de l'exemple de programme Get**

Le programme ouvre la file d'attente cible pour l'obtention des messages ; il utilise l'appel MQOPEN avec l'option OOINPQ. S'il ne parvient pas à ouvrir la file d'attente, le programme écrit un message d'erreur contenant le code anomalie renvoyé par l'appel MQOPEN dans son fichier spoule.

Pour chaque message de la file d'attente, le programme utilise l'appel MQGET pour supprimer le message de la file d'attente ; il affiche ensuite les données contenues dans le message. L'appel MQGET utilise l'option GMWT, en spécifiant un intervalle d'attente (*GMWT*) de 15 secondes, de sorte que le programme attend pendant cette période s'il n'y a pas de message dans la file d'attente. Si aucun message n'arrive avant l'expiration de cet intervalle, l'appel échoue et renvoie le code anomalie RC2033 (aucun message disponible).

Le programme montre comment vous devez effacer les zones *MDMID* et *MDCID* de la structure *MQMD* après chaque appel *MQGET* car l'appel définit ces zones sur les valeurs contenues dans le message qu'il extrait. L'effacement de ces zones signifie que les appels *MQGET* successifs extraient les messages dans l'ordre dans lequel ils sont placés dans la file d'attente.

L'appel *MQGET* spécifie une mémoire tampon de taille fixe. Si un message est plus long que cette mémoire tampon, l'appel échoue et le programme s'arrête.

Le programme se poursuit jusqu'à ce que l'appel *MQGET* renvoie le code anomalie *RC2033* (aucun message disponible) ou que l'appel *MQGET* échoue. Si l'appel échoue, le programme affiche un message d'erreur contenant le code anomalie.

Le programme ferme ensuite la file d'attente à l'aide de l'appel *MQCLOSE*.

### **Exemple de programme de demande sous IBM i**

L'exemple de programme de demande, *AMQ3REQ4*, illustre le traitement client/serveur. Il s'agit du client qui insère des messages de demande dans une file d'attente traitée par un programme serveur. Il attend que le programme serveur place un message de réponse dans une file d'attente de réponse.

L'exemple Demande insère une série de messages de demande dans une file d'attente à l'aide de l'appel *MQPUT*. Ces messages indiquent *SYSTEM.SAMPLE.REPLY* comme file d'attente de réponse. Le programme attend les messages de réponse, puis les affiche. Les réponses sont envoyées uniquement si la file d'attente cible (que nous appellerons la *file d'attente du serveur*) est en cours de traitement par une application serveur ou si une application est déclenchée à cette fin (les exemples de programme *Inquire* et *Set* sont conçus pour être déclenchés). L'exemple attend 5 minutes que la première réponse arrive (pour permettre le déclenchement d'une application serveur) et 15 secondes pour les réponses suivantes, mais il peut s'arrêter sans obtenir de réponse.

Pour démarrer le programme, appelez le programme et indiquez le nom de votre file d'attente cible comme paramètre de programme. Le programme place un ensemble de messages fixes dans la file d'attente ; ces messages sont extraits du bloc de données à la fin du code source du programme.

### **Conception de l'exemple de programme de demande**

Le programme ouvre la file d'attente du serveur pour qu'elle puisse insérer des messages. Il utilise l'appel *MQOPEN* avec l'option *OOOUT*. S'il ne parvient pas à ouvrir la file d'attente, le programme affiche un message d'erreur contenant le code anomalie renvoyé par l'appel *MQOPEN*.

Le programme ouvre ensuite la file d'attente de réponses appelée *SYSTEM.SAMPLE.REPLY* pour obtenir des messages de réponse. Pour cela, le programme utilise l'appel *MQOPEN* avec l'option *OOINPX*. S'il ne parvient pas à ouvrir la file d'attente, le programme affiche un message d'erreur contenant le code anomalie renvoyé par l'appel *MQOPEN*.

Pour chaque ligne d'entrée, le programme lit ensuite le texte dans une mémoire tampon et utilise l'appel *MQPUT* pour créer un message de demande contenant le texte de cette ligne. Lors de cet appel, le programme utilise l'option de rapport *ROEXCD* pour demander que tous les messages de rapport envoyés à propos du message de demande incluent les 100 premiers octets des données du message. Le programme se poursuit jusqu'à ce qu'il atteigne la fin de l'entrée ou que l'appel *MQPUT* échoue.

Le programme utilise ensuite l'appel *MQGET* pour supprimer les messages de réponse de la file d'attente et affiche les données contenues dans les réponses. L'appel *MQGET* utilise l'option *GMWT*, en spécifiant un intervalle d'attente (*GMWI*) de 5 minutes pour la première réponse (afin de permettre le déclenchement d'une application serveur) et de 15 secondes pour les réponses suivantes. Le programme attend ces périodes s'il n'y a pas de message dans la file d'attente. Si aucun message n'arrive avant l'expiration de cet intervalle, l'appel échoue et renvoie le code anomalie *RC2033* (aucun message disponible). L'appel utilise également l'option *GMATM*, de sorte que les messages plus longs que la taille de mémoire tampon déclarée sont tronqués.

Le programme montre comment vous devez effacer les zones *MDMID* et *MDCOD* de la structure *MQMD* après chaque appel *MQGET* car l'appel définit ces zones sur les valeurs contenues dans le message qu'il extrait. L'effacement de ces zones signifie que les appels *MQGET* successifs extraient les messages dans l'ordre dans lequel ils sont placés dans la file d'attente.

Le programme se poursuit jusqu'à ce que l'appel MQGET renvoie le code anomalie RC2033 (aucun message disponible) ou que l'appel MQGET échoue. Si l'appel échoue, le programme affiche un message d'erreur contenant le code anomalie.

Le programme ferme ensuite la file d'attente du serveur et la file d'attente de réponse à l'aide de l'appel MQCLOSE. Le Tableau 813, à la page 1492 présente les modifications apportées à l'exemple de programme Echo qui sont nécessaires pour exécuter les exemples de programme Inquire and Set.

**Remarque :** Les détails de l'exemple de programme Echo sont inclus comme référence.

Tableau 813. Détails de l'exemple de programme client-serveur		
Nom du programme	File d'attente SYSTEM/SAMPLE	Le programme a démarré
Echo	echo	AMQ3ECH4
Interrogation	INQ	AMQ3INQ4
Définition	SET	AMQ3SET4

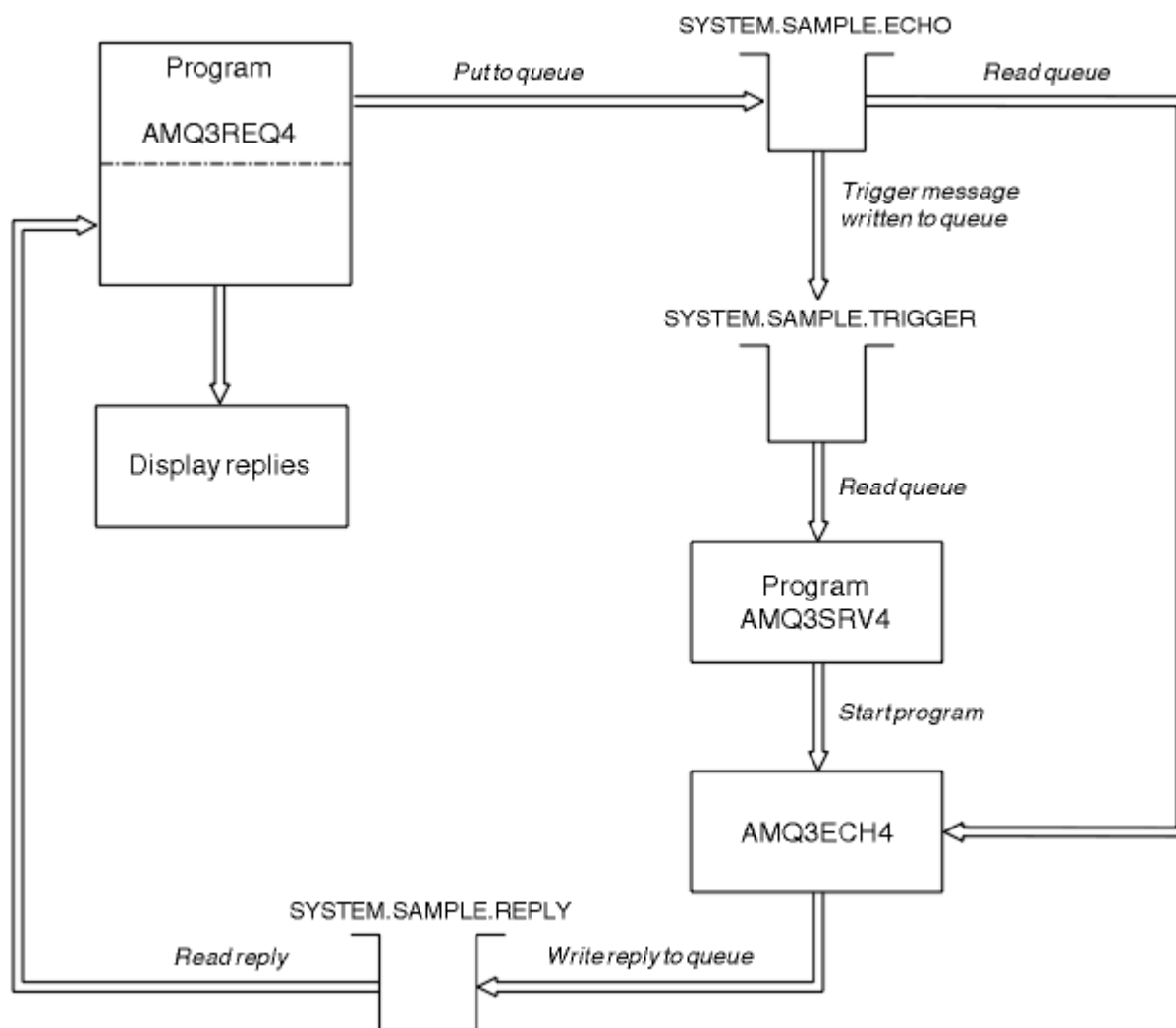


Figure 9. Exemple de diagramme de programme client-serveur (Echo)

Pour exécuter l'exemple à l'aide du déclenchement, démarrez le programme de serveur de déclenchement, AMQ3SRV4, sur la file d'attente d'initialisation requise dans un travail, puis démarrez AMQ3REQ4 dans un autre travail.

Cela signifie que le serveur de déclenchement est prêt lorsque l'exemple de programme de demande envoie un message.

**Remarque :**

1. Les exemples utilisent la file d'attente SYSTEM SAMPLE TRIGGER comme file d'attente d'initialisation pour SYSTEM.SAMPLE.ECHO, SYSTEM.SAMPLE.INQ ou SYSTEM.SAMPLE.SET files d'attente locales. Vous pouvez également définir votre propre file d'attente d'initialisation.
2. Les exemples de définition créés par AMQSAMP4 déclenchent la version C de l'exemple. Si vous souhaitez déclencher la version RPG, vous devez modifier les définitions de processus SYSTEM.SAMPLE.ECHOPROCESS et SYSTEM.SAMPLE.INQPROCESS et SYSTEM.SAMPLE.SETPROCESS. Vous pouvez utiliser la commande CHGMQMPRC (voir [Change MQ Process \(CHGMQMPRC\)](#) pour plus de détails) ou éditer et exécuter votre propre version de AMQSAMP4.
3. Vous devez compiler le programme du serveur de déclenchement à partir de la source fournie dans QMQMSAMP/QRPGLESRC.

Selon le processus de déclenchement que vous souhaitez exécuter, AMQ3REQ4 doit être appelé avec le paramètre spécifiant les messages de demande à placer dans l'une des files d'attente de serveur d'exemples suivantes:

- SYSTEM.SAMPLE.ECHO (pour les exemples de programmes Echo)
- SYSTEM.SAMPLE.INQ (pour les exemples de programmes Inquire)
- SYSTEM.SAMPLE.SET (pour les exemples de programme Set)

Un diagramme de flux pour SYSTEM.SAMPLE.ECHO est présenté dans la [Figure 9](#), à la [page 1492](#). A l'aide de l'exemple, la commande permettant d'émettre la demande de programme RPG sur ce serveur est la suivante:

```
CALL PGM(QMQMSAMP/AMQ3REQ4) PARM('SYSTEM.SAMPLE.ECHO
+ 30 blank characters','Queue_Manager_Name')
```

car le nom de la file d'attente et le nom du gestionnaire de files d'attente doivent comporter 48 caractères.

**Remarque :** Cet exemple de file d'attente a un type de déclencheur FIRST. Par conséquent, s'il existe déjà des messages dans la file d'attente avant l'exécution de l'exemple Demande, les applications serveur ne sont pas déclenchées par les messages que vous envoyez.

Si vous souhaitez essayer d'autres exemples, vous pouvez essayer les variantes suivantes:

- Utilisez AMQ3TRG4 à la place de AMQ3SRV4 pour soumettre le travail à la place, mais les retards potentiels de soumission de travail pourraient rendre plus difficile le suivi de ce qui se passe.
- Utilisez SYSTEM.SAMPLE.INQ et SYSTEM.SAMPLE.SET . A l'aide de l'exemple de fichier de données, les commandes permettant d'émettre les demandes de programme RPG vers ces serveurs sont les suivantes:

```
CALL PGM(QMQMSAMP/AMQ3INQ4) PARM('SYSTEM.SAMPLE.INQ
+ 31 blank characters')
CALL PGM(QMQMSAMP/AMQ3SET4) PARM('SYSTEM.SAMPLE.SET
+ 31 blank characters')
```

car le nom de la file d'attente doit comporter 48 caractères.

Ces exemples de files d'attente ont également un type de déclencheur FIRST.

## **Exemple de programme Echo sur IBM i**

Les exemples de programme Echo renvoient le message envoyé à une file d'attente de réponses. Le programme est nommé AMQ3ECH4

Pour que le processus de déclenchement fonctionne, vous devez vous assurer que l'exemple de programme Echo que vous souhaitez utiliser est déclenché par les messages arrivant dans la file d'attente SYSTEM.SAMPLE.ECHO. Pour ce faire, indiquez le nom de l'exemple de programme Echo à utiliser dans la zone *AppId* de la définition de processus SYSTEM.SAMPLE.ECHOPROCESS. (Pour cela, vous pouvez utiliser la commande CHGMQMPRC, décrite dans *Administration d' IBM i*.) Le type de déclencheur de l'exemple de file d'attente est FIRST. Par conséquent, s'il existe déjà des messages dans la file d'attente avant l'exécution de l'exemple Demande, l'exemple Echo n'est pas déclenché par les messages que vous envoyez.

Une fois la définition correctement définie, démarrez d'abord AMQ3SRV4 dans un travail, puis démarrez AMQ3REQ4 dans un autre travail. Vous pouvez utiliser AMQ3TRG4 au lieu de AMQ3SRV4, mais les retards potentiels de soumission des travaux peuvent rendre plus difficile le suivi de ce qui se passe.

Utilisez les exemples de programmes de demande pour envoyer des messages à la file d'attente SYSTEM.SAMPLE.ECHO. Les exemples de programme Echo envoient un message de réponse contenant les données du message de demande à la file d'attente de réponse indiquée dans le message de demande.

## **Conception de l'exemple de programme Echo**

Lorsque le programme est déclenché, il se connecte explicitement au gestionnaire de files d'attente par défaut à l'aide de l'appel MQCONN. Bien que cela ne soit pas nécessaire sous IBM i, cela signifie que vous pouvez utiliser le même programme sur d'autres plateformes sans changer le code source.

Le programme ouvre ensuite la file d'attente indiquée dans la structure de message de déclenchement qu'il a transmise lors de son démarrage. (Pour plus de clarté, nous appellerons cela la *file d'attente des demandes*.) Le programme utilise l'appel MQOPEN pour ouvrir cette file d'attente pour l'entrée partagée.

Le programme utilise l'appel MQGET pour supprimer des messages de cette file d'attente. Cet appel utilise les options GMATM et GMWT, avec un intervalle d'attente de 5 secondes. Le programme teste le descripteur de chaque message pour voir s'il s'agit d'un message de demande ; si ce n'est pas le cas, le programme supprime le message et affiche un message d'avertissement.

Pour chaque message de demande supprimé de la file d'attente des demandes, le programme utilise l'appel MQPUT pour placer un message de réponse dans la file d'attente des réponses. Ce message contient le contenu du message de demande.

Lorsqu'il ne reste aucun message dans la file d'attente des demandes, le programme ferme cette file d'attente et se déconnecte du gestionnaire de files d'attente.

Ce programme peut également répondre aux messages envoyés à la file d'attente à partir de plateformes autres que IBM i, bien qu'aucun exemple ne soit fourni pour cette situation. Pour que le programme ECHO fonctionne, vous devez :

- Ecrivez un programme en spécifiant correctement les zones *Format*, *Encoding* et *CCSID* pour envoyer des messages de demande de texte.

Le programme ECHO demande au gestionnaire de files d'attente d'effectuer la conversion des données de message, si nécessaire.

- Indiquez CONVERT (\*YES) sur le canal émetteur IBM MQ for IBM i si le programme que vous avez écrit ne fournit pas de conversion similaire pour la réponse.

## **Exemple de programme Inquire sur IBM i**

L'exemple de programme d'interrogation, AMQ3INQ4, demande certains des attributs d'une file d'attente à l'aide de l'appel MQINQ.

Le programme étant destiné à être exécuté en tant que programme déclenché, sa seule entrée est une structure MQTMC (message de déclenchement). Cette structure contient le nom d'une file d'attente cible dont les attributs doivent être renseignés.

Pour que le processus de déclenchement fonctionne, vous devez vous assurer que l'exemple de programme Inquire est déclenché par les messages arrivant dans la file d'attente SYSTEM.SAMPLE.INQ. Pour cela, indiquez le nom de l'exemple de programme Inquire dans la zone *AppId* de SYSTEM.SAMPLE.INQPROCESS . (Pour cela, vous pouvez utiliser la commande CHGMQMPCR, décrite dans [Modification du processus MQ \(CHGMQMPCR\)](#) ). Le type de déclencheur de l'exemple de file d'attente est FIRST. Par conséquent, s'il y a déjà des messages dans la file d'attente avant l'exécution de l'exemple de demande, l'exemple Inquire n'est pas déclenché par les messages que vous envoyez.

Une fois la définition correctement définie, démarrez d'abord AMQ3SRV4 dans un travail, puis démarrez AMQ3REQ4 dans un autre travail. Vous pouvez utiliser AMQ3TRG4 au lieu de AMQ3SRV4, mais les retards potentiels de soumission des travaux peuvent rendre plus difficile le suivi de ce qui se passe.

Utilisez l'exemple de programme de demande pour envoyer des messages de demande, chacun contenant uniquement un nom de file d'attente, à la file d'attente SYSTEM.SAMPLE.INQ. Pour chaque message de demande, l'exemple de programme Inquire envoie un message de réponse contenant des informations sur la file d'attente indiquée dans le message de demande. Les réponses sont envoyées à la file d'attente de réponse indiquée dans le message de demande.

## Conception de l'exemple de programme Inquire

Lorsque le programme est déclenché, il se connecte explicitement au gestionnaire de files d'attente par défaut à l'aide de l'appel MQCONN. Bien que cela ne soit pas nécessaire sous IBM i, cette fonction de conception signifie que vous pouvez utiliser le même programme sur d'autres plateformes sans modifier le code source.

Le programme ouvre ensuite la file d'attente indiquée dans la structure de message de déclenchement qu'il a transmise lors de son démarrage. (Pour plus de clarté, nous appellerons cela la *file d'attente des demandes*.) Le programme utilise l'appel MQOPEN pour ouvrir cette file d'attente pour l'entrée partagée.

Le programme utilise l'appel MQGET pour supprimer des messages de cette file d'attente. Cet appel utilise les options GMATM et GMWT, avec un intervalle d'attente de 5 secondes. Le programme teste le descripteur de chaque message pour voir s'il s'agit d'un message de demande ; dans le cas contraire, le programme supprime le message et affiche un message d'avertissement.

Pour chaque message de demande supprimé de la file d'attente des demandes, le programme lit le nom de la file d'attente (que nous appellerons la *file d'attente cible* ) contenues dans les données et ouvre cette file d'attente à l'aide de l'appel MQOPEN avec l'option OOINQ. Le programme utilise ensuite l'appel MQINQ pour consulter les valeurs des attributs **InhibitGet**, **CurrentQDepth** et **OpenInputCount** de la file d'attente cible.

Si l'appel MQINQ aboutit, le programme utilise l'appel MQPUT pour placer un message de réponse dans la file d'attente de réponse. Ce message contient les valeurs des trois attributs.

Si l'appel MQOPEN ou MQINQ échoue, le programme utilise l'appel MQPUT pour placer un message *report* dans la file d'attente de réponse. Dans la zone *MDFB* du descripteur de message de ce message de rapport, le code anomalie est renvoyé par l'appel MQOPEN ou MQINQ, selon celui qui a échoué.

Après l'appel MQINQ, le programme ferme la file d'attente cible à l'aide de l'appel MQCLOSE.

Lorsqu'il ne reste aucun message dans la file d'attente des demandes, le programme ferme cette file d'attente et se déconnecte du gestionnaire de files d'attente.

## Exemple de programme Set sous IBM i

L'exemple de programme Set, AMQ3SET4, inhibe les opérations d'insertion dans une file d'attente en utilisant l'appel MQSET pour modifier l'attribut **InhibitPut** de la file d'attente.

Le programme est destiné à être exécuté en tant que programme déclenché, de sorte que sa seule entrée est une structure MQTMC (message de déclenchement) qui contient le nom d'une file d'attente cible avec les attributs qui doivent être renseignés.

Pour que le processus de déclenchement fonctionne, vous devez vous assurer que l'exemple de programme Set est déclenché par les messages arrivant dans la file d'attente SYSTEM.SAMPLE.SET. Pour ce faire, indiquez le nom de l'exemple de programme Set dans la zone *AppLId* de la définition de processus SYSTEM.SAMPLE.SETPROCESS. (Pour cela, vous pouvez utiliser la commande CHGMQMPCRC, décrite dans la rubrique [Administration d' IBM i](#) .) Le type de déclencheur de l'exemple de file d'attente est FIRST. Par conséquent, s'il y a déjà des messages dans la file d'attente avant l'exécution de l'exemple de demande, l'exemple de définition n'est pas déclenché par les messages que vous envoyez.

Une fois la définition correctement définie, démarrez d'abord AMQ3SRV4 dans un travail, puis démarrez AMQ3REQ4 dans un autre travail. Vous pouvez utiliser AMQ3TRG4 au lieu de AMQ3SRV4, mais les retards potentiels de soumission des travaux peuvent rendre plus difficile le suivi de ce qui se passe.

Utilisez l'exemple de programme Demande pour envoyer des messages de demande, chacun contenant uniquement un nom de file d'attente, à la file d'attente SYSTEM.SAMPLE.SET. Pour chaque message de demande, l'exemple de programme Set envoie un message de réponse contenant une confirmation que les opérations d'insertion ont été interdites dans la file d'attente indiquée. Les réponses sont envoyées à la file d'attente de réponse indiquée dans le message de demande.

## Conception de l'exemple de programme Set

Lorsque le programme est déclenché, il se connecte explicitement au gestionnaire de files d'attente par défaut à l'aide de l'appel MQCONN. Bien que cela ne soit pas nécessaire sous IBM i, cela signifie que vous pouvez utiliser le même programme sur d'autres plateformes sans changer le code source.

Le programme ouvre ensuite la file d'attente indiquée dans la structure de message de déclenchement qu'il a transmise lors de son démarrage. (Pour plus de clarté, nous appellerons cela la *file d'attente des demandes*.) Le programme utilise l'appel MQOPEN pour ouvrir cette file d'attente pour l'entrée partagée.

Le programme utilise l'appel MQGET pour supprimer des messages de cette file d'attente. Cet appel utilise les options GMATM et GMWT, avec un intervalle d'attente de 5 secondes. Le programme teste le descripteur de chaque message pour voir s'il s'agit d'un message de demande ; si ce n'est pas le cas, le programme supprime le message et affiche un message d'avertissement.

Pour chaque message de demande supprimé de la file d'attente des demandes, le programme lit le nom de la file d'attente (que nous appellerons la *file d'attente cible* ) contenues dans les données et ouvre cette file d'attente à l'aide de l'appel MQOPEN avec l'option OOSSET. Le programme utilise ensuite l'appel MQSET pour définir la valeur de l'attribut **InhibitPut** de la file d'attente cible sur QAPUTI.

Si l'appel MQSET aboutit, le programme utilise l'appel MQPUT pour placer un message de réponse dans la file d'attente de réponse. Ce message contient la chaîne PUT inhibited.

Si l'appel MQOPEN ou MQSET échoue, le programme utilise l'appel MQPUT pour placer un message *report* dans la file d'attente de réponse. Dans la zone *MDFB* du descripteur de message de ce message de rapport, il s'agit du code anomalie renvoyé par l'appel MQOPEN ou MQSET, selon celui qui a échoué.

Après l'appel MQSET, le programme ferme la file d'attente cible à l'aide de l'appel MQCLOSE.

Lorsqu'il ne reste aucun message dans la file d'attente des demandes, le programme ferme cette file d'attente et se déconnecte du gestionnaire de files d'attente.

## Exemples de programmes de déclenchement sur IBM i

IBM MQ for IBM i fournit deux exemples de programmes de déclenchement écrits en ILE/RPG.

Les programmes sont les suivants:

### AMQ3TRG4

Il s'agit d'un moniteur de déclenchement pour l'environnement IBM i . Il soumet un travail IBM i pour que l'application soit démarrée, ce qui signifie qu'un coût de traitement supplémentaire est associé à chaque message de déclenchement.

### AMQ3SRV4

Il s'agit d'un serveur de déclenchement pour l'environnement IBM i . Pour chaque message de déclenchement, ce serveur exécute la commande de démarrage dans son propre travail pour démarrer l'application spécifiée. Le serveur de déclenchement peut appeler des transactions CICS .



Les versions en langage C de ces exemples sont également disponibles en tant que programmes exécutables dans la bibliothèque QMQM, appelée AMQSTRG4 et AMQSERV4.

#### *Exemple de moniteur de déclenchement AMQ3TRG4 sous IBM i*

AMQ3TRG4 est un moniteur de déclenchement. Il prend un paramètre: le nom de la file d'attente d'initialisation qu'il doit servir. AMQSAMP4 définit un exemple de file d'attente d'initialisation, SYSTEM.SAMPLE.TRIGGER, que vous pouvez utiliser lorsque vous essayez les exemples de programme.

AMQ3TRG4 soumet un travail IBM i pour chaque message de déclenchement valide qu'il extrait de la file d'attente d'initialisation.

## Conception du moniteur de déclenchement

Le moniteur de déclenchement ouvre la file d'attente d'initialisation et extrait les messages de la file d'attente, en spécifiant un intervalle d'attente illimité.

Le moniteur de déclenchement soumet un travail IBM i pour démarrer l'application spécifiée dans le message de déclenchement et transmet une structure MQTMC (version alphanumérique du message de déclenchement). Les données d'environnement du message de déclenchement sont utilisées comme paramètres de soumission de travail.

Enfin, le programme ferme la file d'attente d'initialisation.

#### *Exemple de serveur de déclenchement AMQ3SRV4*

AMQ3SRV4 est un serveur de déclenchement. Il prend un paramètre: le nom de la file d'attente d'initialisation qu'il doit servir. AMQSAMP4 définit un exemple de file d'attente d'initialisation, SYSTEM.SAMPLE.TRIGGER, que vous pouvez utiliser lorsque vous essayez les exemples de programme.

Pour chaque message de déclenchement, AMQ3SRV4 exécute une commande de démarrage dans son propre travail pour démarrer l'application spécifiée.

A l'aide de l'exemple de file d'attente de déclenchement, la commande à émettre est la suivante:

```
CALL PGM(QMQM/AMQ3SRV4) PARM('Queue Name')
```

Où Queue Name doit avoir une longueur de 48 caractères, que vous pouvez obtenir en complétant le nom de la file d'attente avec le nombre requis de blancs. Par conséquent, si vous utilisez SYSTEM.SAMPLE.TRIGGER comme file d'attente cible, vous aurez besoin de 28 caractères blancs.

## Conception du serveur de déclenchement

La conception du serveur de déclenchement est similaire à celle du moniteur de déclenchement, à l'exception du serveur de déclenchement:

- Autorise les applications CICS et IBM i
- N'utilise pas les données d'environnement du message de déclenchement
- Appelle des applications IBM i dans son propre travail (ou utilise STRCICSUSR pour démarrer des applications CICS) au lieu de soumettre un travail IBM i
- Ouvre la file d'attente d'initialisation pour l'entrée partagée, de sorte que de nombreux serveurs de déclenchement puissent s'exécuter en même temps

**Remarque :** Les programmes démarrés par AMQ3SRV4 ne doivent pas utiliser l'appel MQDISC car cela arrêtera le serveur de déclenchement. Si les programmes démarrés par AMQ3SRV4 utilisent l'appel MQCONN, ils obtiennent le code anomalie RC2002 .

#### *Arrêt des exemples de programmes de déclenchement sous IBM i*

Un programme de moniteur de déclenchement peut être arrêté par l'option 2 de sysrequest (ENDRQS) ou en inhibant les extractions de la file d'attente de déclenchement.

Si l'exemple de file d'attente de déclenchement est utilisé, la commande est la suivante:

```
CHGMQM QNAME('SYSTEM.SAMPLE.TRIGGER') GETENBL(*NO)
```

**Remarque :** Pour relancer le déclenchement sur cette file d'attente, vous devez entrer la commande suivante:

```
CHGMQM QNAME('SYSTEM.SAMPLE.TRIGGER') GETENBL(*YES)
```

### ***Exécution des exemples à l'aide de files d'attente distantes sous IBM i***

Vous pouvez illustrer la mise en file d'attente distante en exécutant les exemples sur les gestionnaires de files d'attente de messages connectés.

Le programme AMQSAMP4 fournit une définition locale d'une file d'attente éloignée (SYSTEM.SAMPLE.REMOTE) qui utilise un gestionnaire de files d'attente éloigné nommé OTHER. Pour utiliser cet exemple de définition, remplacez OTHER par le nom du deuxième gestionnaire de files d'attente de messages que vous souhaitez utiliser. Vous devez également configurer un canal de transmission de messages entre vos deux gestionnaires de files d'attente de messages. Pour plus d'informations sur cette opération, voir [Programmes d'exit de canal pour les canaux de messagerie](#).

L'exemple de programme de demande insère son propre nom de gestionnaire de files d'attente local dans la zone *MDRM* des messages qu'il envoie. Les exemples d'interrogation et d'interrogation envoient des messages de réponse à la file d'attente et au gestionnaire de files d'attente de messages nommés dans les zones *MDRQ* et *MDRM* des messages de demande qu'ils traitent.

## **Codes retour pour IBM i (ILE RPG)**

Ces informations décrivent les codes retour associés à MQI et MQAI.

Les codes retour associés à:

- Les commandes PCF (Programmable Command Format) sont répertoriées dans le [Informations de référence sur les formats de commande programmables](#).
- Les appels C++ sont répertoriés dans [Utilisation de C++](#).

Pour chaque appel, un code achèvement et un code raison sont renvoyés par le gestionnaire de files d'attente ou par une routine d'exit, pour indiquer la réussite ou l'échec de l'appel.

Les applications ne doivent pas dépendre d'erreurs vérifiées dans un ordre spécifique, sauf lorsqu'elles sont spécifiquement mentionnées. Si plusieurs codes achèvement ou codes raison peuvent provenir d'un appel, l'erreur particulière signalée dépend de l'implémentation.

## **Codes achèvement pour IBM i (ILE RPG)**

Le paramètre de code achèvement (*CMPCOD*) permet à l'appelant de voir rapidement si l'appel a abouti, s'est terminé partiellement ou a échoué.

### **CCOK**

(MQCC\_OK sur d'autres plateformes)

Achèvement réussi.

L'appel a abouti ; tous les paramètres de sortie ont été définis. Le paramètre **REASON** a toujours la valeur RCNONE dans ce cas.

### **CCWARN**

(MQCC\_WARN sur d'autres plateformes)

Avertissement (achèvement partiel).

L'appel s'est terminé partiellement. Certains paramètres de sortie peuvent avoir été définis en plus des paramètres de sortie *CMPCOD* et *REASON*. Le paramètre **REASON** fournit des informations supplémentaires sur l'exécution partielle.

### **CCFAIL**

(MQCC\_FAIL sur d'autres plateformes)

Echec de l'appel.

Le traitement de l'appel n'a pas abouti et l'état du gestionnaire de files d'attente est normalement inchangé ; des exceptions sont spécifiquement notées. Les paramètres de sortie *CMPCOD* et *REASON* ont été définis ; les autres paramètres sont inchangés, sauf indication contraire.

Il peut s'agir d'une erreur dans le programme d'application ou d'une situation externe au programme. Par exemple, les droits de l'utilisateur peuvent avoir été révoqués. Le paramètre **REASON** fournit des informations supplémentaires sur l'erreur.

## **Codes raison pour IBM i (ILE RPG)**

Le paramètre de code anomalie (*REASON*) est une qualification au paramètre de code achèvement (*CMPCOD*).

S'il n'y a pas de raison particulière à signaler, RCNONE est renvoyé. Un appel réussi renvoie CCOK et RCNONE.

Si le code achèvement est CCWARN ou CCFAIL, le gestionnaire de files d'attente indique toujours une raison de qualification ; des détails sont fournis sous chaque description d'appel.

Lorsque des routines d'exit utilisateur définissent des codes achèvement et des raisons, elles doivent respecter ces règles. En outre, les valeurs de raison spéciale définies par les exits utilisateur doivent être inférieures à zéro, afin de garantir qu'elles n'entrent pas en conflit avec les valeurs définies par le gestionnaire de files d'attente. Les exits peuvent définir des raisons déjà définies par le gestionnaire de files d'attente, lorsqu'elles sont appropriées.

Les codes anomalie apparaissent également dans :

- Zone *DLREA* de la structure MQDLH
- Zone *MDFB* de la structure MQMD

Pour obtenir la liste complète des codes anomalie, voir [Codes anomalie et achèvement d'API](#).

Pour trouver votre code anomalie IBM i dans cette liste, supprimez le code "RC" de l'avant, par exemple RC2002 devient 2002. Les codes achèvement sont également affichés tels qu'ils sont sur d'autres plateformes :

<b>IBM i</b>	<b>Autres plateformes</b>
CCOK	MQCC_OK
CCWARN	AVERTISSEMENT MQCC
CCFAIL	MQCC_ECHEC

## **Règles de validation des options MQI pour IBM i (ILE RPG)**

Cette rubrique fournit des informations sur les situations qui produisent un code anomalie RC2046 à partir d'un appel MQOPEN, MQPUT, MQPUT1, MQGET ou MQCLOSE.

### **Appel MQOPEN sur IBM i**

Pour les options de l'appel MQOPEN :

- *Au moins un* des éléments suivants doit être spécifié :

- OOB<sub>BRW</sub>
- OOIN<sub>PQ</sub>
- OOIN<sub>PX</sub>
- OOIN<sub>PS</sub>
- OOIN<sub>Q</sub>
- OOOUT
- OOSET
- Seul *un* des éléments suivants est autorisé:
  - OOIN<sub>PQ</sub>
  - OOIN<sub>PX</sub>
  - OOIN<sub>PS</sub>
- Seul *un* des éléments suivants est autorisé:
  - OOB<sub>NDO</sub>
  - Nom distinctif OBN<sub>DN</sub>
  - OOB<sub>NDQ</sub>

**Remarque :** Les options répertoriées précédemment s'excluent mutuellement. Toutefois, comme la valeur de OOB<sub>NDQ</sub> est égale à zéro, le fait de la spécifier avec l'une des deux autres options de liaison n'entraîne pas le code anomalie RC2046. OOB<sub>NDQ</sub> est fourni à la documentation du programme d'aide.

- Si OOSAVA est spécifié, l'une des options OOIN<sub>P\*</sub> doit également être spécifiée.
- Si l'une des options OOSET<sub>\*</sub> ou OOPAS<sub>\*</sub> est spécifiée, OOOUT doit également être spécifié.

## Appel MQPUT sur IBM i

Pour les options d'insertion de message:

- La combinaison de PMSYP et de PMNSYP n'est pas admise.
- Seul *un* des éléments suivants est autorisé:
  - PMDEF<sub>C</sub>
  - PMNO<sub>C</sub>
  - PMPAS<sub>A</sub>
  - PMPAS<sub>I</sub>
  - PMSET<sub>A</sub>
  - PMSET<sub>I</sub>
- PMALTU n'est pas autorisé (il est valide uniquement sur l'appel MQPUT1).

## Appel MQPUT1 sur IBM i

Pour les options d'insertion de message, les règles sont les mêmes que pour l'appel MQPUT, à l'exception des options suivantes:

- PMALTU est autorisé.
- PMLOGO n'est pas autorisé.

## Appel MQGET sur IBM i

Pour les options get-message:

- Seule *une* des options suivantes est autorisée:
  - GMNSYP

- GMSYP
- GMPSYP
- Seule *une* des options suivantes est autorisée:
  - MGMBRWF
  - GMBRWC
  - GMBRWN
  - UC de la GMMUC
- GMSYP n'est pas autorisé avec l'une des options suivantes:
  - MGMBRWF
  - GMBRWC
  - GMBRWN
  - GMLK
  - GMUNLK
- GMPSYP n'est pas autorisé avec l'une des options suivantes:
  - MGMBRWF
  - GMBRWC
  - GMBRWN
  - GMCMPM
  - GMUNLK
- Si GMLK est spécifié, l'une des options suivantes doit également être spécifiée:
  - MGMBRWF
  - GMBRWC
  - GMBRWN
- Si GMUNLK est spécifié, seules les options suivantes sont autorisées:
  - GMNSYP
  - GMNWT

### **Appel MQCLOSE sur IBM i**

- Pour les options de l'appel MQCLOSE. La combinaison de CODEL et COPURG n'est pas autorisée.
- Une seule des options suivantes est autorisée:
  - COKPSB
  - CORMSB

### **Appel MQSUB sur IBM i**

Pour les options de l'appel MQSUB:

- Au moins un des éléments suivants doit être spécifié:
- Au moins un des éléments suivants doit être spécifié:
  - SOALT
  - SORES
  - SOCRT
- Une seule des options suivantes est autorisée:
  - SODUR

– SONDUR

**Remarque :** Les options répertoriées précédemment s'excluent mutuellement. Cependant, comme la valeur de SONDUR est égale à zéro, le fait de la spécifier avec SODUR ne génère pas le code anomalie RC2046. SONDUR est fourni pour aider la documentation du programme.

- La combinaison de SOGRP et de SOMAN n'est pas autorisée.
- SOGRP requiert la spécification de SOSCID.
- Une seule des options suivantes est autorisée: SOAUID SOFUID
- La combinaison de SONEWP et de SOPUBR n'est pas autorisée.
- SONEWP est uniquement autorisé en combinaison avec SOCRT.
- Une seule des options suivantes est autorisée:
  - SOWCHR (RSC)
  - SOWTOP

## Codages de machine sur IBM i

Utilisez ces informations pour en savoir plus sur la structure de la zone *MDENC* dans le descripteur de message.

Pour plus d'informations sur le descripteur de message, voir [«MQMD \(descripteur de message\) sous IBM i», à la page 1162.](#)

La zone *MDENC* est un entier 32 bits qui est divisé en quatre sous-zones distinctes ; ces sous-zones identifient:

- Codage utilisé pour les entiers binaires
- Codage utilisé pour les entiers décimaux condensés
- Codage utilisé pour les nombres à virgule flottante
- Bits réservés

Chaque sous-champ est identifié par un masque de bit qui a 1 bit dans les positions correspondant au sous-champ, et 0 bit ailleurs. Les bits sont numérotés de sorte que le bit 0 est le bit de poids fort, et le bit 31 le bit de poids faible. Les masques suivants sont définis:

### **ENIMSK**

Masque pour le codage d'entier binaire.

Cette sous-zone occupe les positions de bit 28 à 31 dans la zone *MDENC* .

### **ENDMSK**

Masque pour le codage d'entier décimal condensé.

Cette sous-zone occupe les positions de bit 24 à 27 dans la zone *MDENC* .

### **KMSJ**

Masque pour le codage en virgule flottante.

Cette sous-zone occupe les positions de bit 20 à 23 dans la zone *MDENC* .

### **ENRMSK**

Masque pour les bits réservés.

Cette sous-zone occupe les positions de bit 0 à 19 dans la zone *MDENC* .

## **Codage d'entier binaire sous IBM i**

Valeurs valides pour le codage d'entier binaire.

Les valeurs suivantes sont valides pour le codage d'entier binaire:

### **ENIUND**

Codage d'entier non défini.

Les entiers binaires sont représentés à l'aide d'un codage non défini.

#### **ENCRS**

Codage d'entier normal.

Les entiers binaires sont représentés de manière classique:

- L'octet le moins significatif du nombre a l'adresse la plus élevée de tous les octets du nombre ; l'octet le plus significatif a l'adresse la plus faible.
- Le bit de poids faible de chaque octet est à côté de l'octet avec l'adresse supérieure suivante ; le bit de poids fort de chaque octet est à côté de l'octet avec l'adresse inférieure suivante.

#### **ENIREV**

Codage d'entier inversé.

Les entiers binaires sont représentés de la même manière qu'ENINOR, mais avec les octets disposés dans l'ordre inverse. Les bits à l'intérieur de chaque octet sont disposés de la même manière qu'ENINOR.

### **IBM i Codage d'entier décimal condensé sous IBM i**

Valeurs valides pour le codage d'entier décimal condensé

Les valeurs suivantes sont valides pour le codage d'entier décimal condensé:

#### **ENDUND**

Codage décimal condensé non défini.

Les entiers décimaux condensés sont représentés à l'aide d'un codage non défini.

#### **ENDNOR**

Codage décimal condensé normal.

Les entiers décimaux condensés sont représentés de manière conventionnelle:

- Chaque chiffre décimal dans la forme imprimable du nombre est représenté en décimal condensé par un chiffre hexadécimal unique compris entre X' 0' et X' 9'. Chaque chiffre hexadécimal occupe 4 bits, de sorte que chaque octet du nombre décimal condensé représente deux chiffres décimaux dans la forme imprimable du nombre.
- L'octet le moins significatif du nombre décimal condensé est l'octet qui contient le chiffre décimal le moins significatif. Dans cet octet, les 4 bits de poids fort contiennent le chiffre décimal de poids faible, et les 4 bits de poids faible contiennent le signe. Le signe est soit X'C'(positif), X'D'(négatif), soit X'F'(non signé).
- L'octet le moins significatif du nombre a l'adresse la plus élevée de tous les octets du nombre ; l'octet le plus significatif a l'adresse la plus faible.
- Le bit de poids faible de chaque octet est à côté de l'octet avec l'adresse supérieure suivante ; le bit de poids fort de chaque octet est à côté de l'octet avec l'adresse inférieure suivante.

#### **REVFIN**

Codage décimal condensé inversé.

Les entiers décimaux condensés sont représentés de la même manière que ENDNOR, mais avec les octets organisés dans l'ordre inverse. Les bits à l'intérieur de chaque octet sont disposés de la même manière que ENDNOR.

### **IBM i Codage à virgule flottante sous IBM i**

Valeurs valides pour le codage en virgule flottante

Les valeurs admises pour le codage en virgule flottante sont les suivantes:

#### **ENFUND**

Codage en virgule flottante non défini.

Les nombres à virgule flottante sont représentés à l'aide d'un codage non défini.

### **ENFNOR**

Normal IEEE (The Institute of Electrical and Electronics Engineers): codage à floteur.

Les nombres à virgule flottante sont représentés à l'aide du format à virgule flottante IEEE standard, avec les octets organisés comme suit:

- L'octet de poids faible de la mantisse a l'adresse la plus élevée de tous les octets du nombre ; l'octet contenant l'exposant a l'adresse la plus faible
- Le bit de poids faible de chaque octet est à côté de l'octet avec l'adresse supérieure suivante ; le bit de poids fort de chaque octet est à côté de l'octet avec l'adresse inférieure suivante

Des détails sur le codage à virgule flottante IEEE peuvent être trouvés dans la norme IEEE 754.

### **ENFREV**

Codage à virgule flottante IEEE inversé.

Les nombres à virgule flottante sont représentés de la même manière que ENFNOR, mais avec les octets disposés dans l'ordre inverse. Les bits à l'intérieur de chaque octet sont disposés de la même manière que ENFNOR.

### **ENF390**

Codage flottant de l'architecture System/390 .

Les nombres à virgule flottante sont représentés à l'aide du format à virgule flottante System/390 standard ; ce format est également utilisé par System/370.

## **IBM i Construction de codages sur IBM i**

Pour construire une valeur pour la zone *MDENC* dans MQMD, vous devez ajouter les constantes appropriées qui décrivent les codages requis.

Veillez à ne combiner qu'un seul des codages ENI\* avec l'un des codages END\* et l'un des codages ENF\*.

## **IBM i Analyse des codages sur IBM i**

La zone *MDENC* contient des sous-zones ; pour cette raison, les applications qui doivent examiner le codage de type entier, décimal condensé ou flottant doivent utiliser la technique décrite dans cette rubrique.

### **Utilisation de l'arithmétique**

Les étapes suivantes doivent être effectuées à l'aide de l'arithmétique entière:

1. Sélectionnez l'une des valeurs suivantes, en fonction du type de codage requis:

- 1 pour le codage d'entier binaire
- 16 pour le codage d'entier décimal condensé
- 256 pour le codage à virgule flottante

Appelez la valeur A.

2. Divisez la valeur de la zone *MDENC* par A ; appeler le résultat B.

3. Divisez B par 16 ; appelez le résultat C.

4. Multipliez C par 16 et soustrayez de B ; appeler le résultat D.

5. Multipliez D par A ; appeler le résultat E.

6. E est le codage requis et peut être testé pour l'égalité avec chacune des valeurs valides pour ce type de codage.



Tableau récapitulant les codages des architectures de machine.

Les codages des architectures de machine sont présentés dans la [Tableau 815](#), à la page 1505.

*Tableau 815. Récapitulatif des codages pour les architectures de machine*

Architecture de la machine	Codage d'entier binaire	Codage d'entier décimal condensé	Codage à virgule flottante
IBM i	normal	normal	IEEE normal
Intel x86	reversed	reversed	IEEE inversé
PowerPC	normal	normal	IEEE normal
System/390	normal	normal	System/390

## Options de rapport et indicateurs de message sous IBM i

Cette rubrique concerne les zones *MDREP* et *MDMFL* qui font partie du descripteur de message MQMD spécifié dans les appels MQGET, MQPUT et MQPUT1 .

Pour plus d'informations sur le descripteur de message, voir «MQMD (descripteur de message) sous IBM i», à la page 1162. Ces informations décrivent:

- Structure de la zone de rapport et mode de traitement du gestionnaire de files d'attente
- Comment une application doit analyser la zone de rapport
- Structure de la zone d'indicateurs de message

### Structure de la zone de rapport

La zone *MDREP* est un entier 32 bits qui est divisé en trois sous-zones distinctes.

Ces sous-zones identifient:

- Options de rapport rejetées si le gestionnaire de files d'attente local ne les reconnaît pas
- Options de rapport toujours acceptées, même si le gestionnaire de files d'attente local ne les reconnaît pas
- Options de rapport acceptées uniquement si certaines autres conditions sont remplies

Chaque sous-champ est identifié par un masque de bit qui a 1 bit dans les positions correspondant au sous-champ, et 0 bit ailleurs. Notez que les bits d'une sous-zone ne sont pas nécessairement adjacents. Les bits sont numérotés de sorte que le bit 0 est le bit de poids fort, et le bit 31 le bit de poids faible. Les masques suivants sont définis pour identifier les sous-zones:

#### RORUM

Masque les options de rapport non prises en charge qui sont rejetées.

Ce masque identifie les positions de bit dans la zone *MDREP* où les options de rapport qui ne sont pas prises en charge par le gestionnaire de files d'attente local entraînent l'échec de l'appel MQPUT ou MQPUT1 avec le code achèvement CCFAIL et le code anomalie RC2061.

Cette sous-zone occupe les positions de bit 3, et 11 à 13.

#### ROAUM

Masque pour les options de rapport non prises en charge qui sont acceptées.

Ce masque identifie les positions de bit dans la zone *MDREP* où les options de rapport qui ne sont pas prises en charge par le gestionnaire de files d'attente local seront néanmoins acceptées sur les appels MQPUT ou MQPUT1 . Le code achèvement CCWARN avec le code anomalie RC2104 est renvoyé dans ce cas.

Cette sous-zone occupe les positions de bit 0 à 2, 4 à 10 et 24 à 31.

Les options de rapport suivantes sont incluses dans cette sous-zone:

- ROCMTC
- RODLQ
- RODISC
- ROEXC
- ROEXCD
- ROEXCF
- ROEXP
- ROEXPD
- ROEXPF
- RONAN
- RONMI
- RONONE
- ROPAN
- ROPCI
- ROPMI

#### **ROAUXM**

Masque les options de rapport non prises en charge qui ne sont acceptées que dans certaines circonstances.

Ce masque identifie les positions de bit dans la zone *MDREP* où les options de rapport qui ne sont pas prises en charge par le gestionnaire de files d'attente local seront néanmoins acceptées sur les appels MQPUT ou MQPUT1 à condition que les deux conditions suivantes soient remplies:

- Le message est destiné à un gestionnaire de files d'attente éloignées.
- L'application ne place pas le message directement dans une file d'attente de transmission locale (c'est-à-dire que la file d'attente identifiée par les zones *ODMN* et *ODON* dans le descripteur d'objet spécifié dans l'appel MQOPEN ou MQPUT1 n'est pas une file d'attente de transmission locale).

Le code achèvement CCWARN avec le code anomalie RC2104 est renvoyé si ces conditions sont satisfaites, et CCFAIL avec le code anomalie RC2061 si ce n'est pas le cas.

Cette sous-zone occupe les positions de bit 14 à 23.

Les options de rapport suivantes sont incluses dans cette sous-zone:

- ROCOA
- ROCOAD
- ROCOAF
- ROCOD
- ROCODD
- ROCODF

S'il existe des options spécifiées dans la zone *MDREP* que le gestionnaire de files d'attente ne reconnaît pas, le gestionnaire de files d'attente vérifie chaque sous-zone à l'aide de l'opération bit à bit AND pour combiner la zone *MDREP* avec le masque de cette sous-zone. Si le résultat de cette opération est différent de zéro, le code achèvement et les codes raison décrits précédemment sont renvoyés.

Si CCWARN est renvoyé, il n'est pas défini quel code anomalie est renvoyé si d'autres conditions d'avertissement existent.

La possibilité de spécifier et d'accepter des options de rapport qui ne sont pas reconnues par le gestionnaire de files d'attente local est utile lorsqu'il est nécessaire d'envoyer un message avec une option de rapport qui sera reconnue et traitée par un gestionnaire de files d'attente *distant*.

## **Analyse de la zone de rapport sur IBM i**

La zone MDREP contient des sous-zones. Pour cette raison, certaines applications doivent vérifier si l'expéditeur du message a demandé un rapport particulier. Ces applications doivent utiliser la technique décrite dans cette rubrique.

### **Utilisation de l'arithmétique**

Les étapes suivantes doivent être effectuées à l'aide de l'arithmétique entière:

1. Sélectionnez l'une des valeurs suivantes, en fonction du type de rapport à vérifier:

- Rapport ROCOA pour COA
- Rapport ROCOD pour COD
- ROEXC pour le rapport d'exception
- ROEXP pour le rapport d'expiration

Appelez la valeur A.

2. Divisez la zone MDREP par A ; appeler le résultat B.

3. Divisez B par 8 ; appeler le résultat C.

4. Multipliez C par 8 et soustrayez de B ; appeler le résultat D.

5. Multipliez D par A ; appeler le résultat E.

6. Testez E pour l'égalité avec chacune des valeurs possibles pour ce type de rapport.

Par exemple, si A est ROEXC, testez E pour l'égalité avec chacun des éléments suivants afin de déterminer ce qui a été spécifié par l'expéditeur du message:

- RONONE
- ROEXC
- ROEXCD
- ROEXCF

Les tests peuvent être effectués dans l'ordre qui convient le mieux à la logique de l'application.

Le pseudocode suivant illustre cette technique pour les messages de rapport d'exception:

```
A = ROEXC
B = Report/A
C = B/8
D = B - C*8
E = D*A
```

Une méthode similaire peut être utilisée pour tester les options ROPMI ou ROPCI ; sélectionnez comme valeur A celle de ces deux constantes qui est appropriée, puis procédez comme décrit précédemment, mais remplacez la valeur 8 dans les étapes précédentes par la valeur 2.

## **Structure de la zone d'indicateurs de message sur IBM i**

La zone MDMFL est un entier 32 bits qui est divisé en trois sous-zones distinctes.

Ces sous-zones identifient:

- Indicateurs de message rejetés si le gestionnaire de files d'attente local ne les reconnaît pas
- Indicateurs de message toujours acceptés, même si le gestionnaire de files d'attente local ne les reconnaît pas

- Indicateurs de message acceptés uniquement si certaines autres conditions sont remplies

**Remarque :** Toutes les sous-zones de *MDMFL* sont réservées pour être utilisées par le gestionnaire de files d'attente.

Chaque sous-champ est identifié par un masque de bit qui a 1 bit dans les positions correspondant au sous-champ, et 0 bit ailleurs. Les bits sont numérotés de sorte que le bit 0 est le bit de poids fort, et le bit 31 le bit de poids faible. Les masques suivants sont définis pour identifier les sous-zones:

#### **MFRUM**

Masque pour les indicateurs de message non pris en charge qui sont rejetés.

Ce masque identifie les positions de bit dans la zone *MDMFL* où les indicateurs de message qui ne sont pas pris en charge par le gestionnaire de files d'attente local entraînent l'échec de l'appel MQPUT ou MQPUT1 avec le code achèvement CCFAIL et le code anomalie RC2249.

Cette sous-zone occupe les positions de bit 20 à 31.

Les indicateurs de message suivants sont inclus dans cette sous-zone:

- MFLMIG
- MFLSEG
- MFMIG
- MFSEG
- MFSEGA
- MFSEGI

#### **MFAUM**

Masque pour les indicateurs de message non pris en charge qui sont acceptés.

Ce masque identifie les positions de bit dans la zone *MDMFL* où les indicateurs de message qui ne sont pas pris en charge par le gestionnaire de files d'attente local seront néanmoins acceptés sur les appels MQPUT ou MQPUT1. Le code achèvement est CCOK.

Cette sous-zone occupe les positions de bit 0 à 11.

#### **MFAUXM**

Masque les indicateurs de message non pris en charge qui ne sont acceptés que dans certaines circonstances.

Ce masque identifie les positions de bit dans la zone *MDMFL* où les indicateurs de message qui ne sont pas pris en charge par le gestionnaire de files d'attente local seront néanmoins acceptés sur les appels MQPUT ou MQPUT1 à condition que les deux conditions suivantes soient remplies:

- Le message est destiné à un gestionnaire de files d'attente éloignées.
- L'application ne place pas le message directement dans une file d'attente de transmission locale (c'est-à-dire que la file d'attente identifiée par les zones *ODMN* et *ODON* dans le descripteur d'objet spécifié dans l'appel MQOPEN ou MQPUT1 n'est pas une file d'attente de transmission locale).

Le code achèvement CCOK est renvoyé si ces conditions sont satisfaites, et CCFAIL avec le code anomalie RC2249 si ce n'est pas le cas.

Cette sous-zone occupe les positions de bit 12 à 19.

Si des indicateurs sont spécifiés dans la zone *MDMFL* que le gestionnaire de files d'attente ne reconnaît pas, le gestionnaire de files d'attente vérifie chaque sous-zone à tour de rôle en utilisant l'opération bit à bit AND pour combiner la zone *MDMFL* avec le masque de cette sous-zone. Si le résultat de cette opération est différent de zéro, le code achèvement et les codes raison décrits précédemment sont renvoyés.

Cette rubrique décrit l'interface de l'exit de conversion de données et le traitement effectué par le gestionnaire de files d'attente lorsque la conversion de données est requise.

L'exit de conversion de données est appelé dans le cadre du traitement de l'appel MQGET. Il est utilisé pour convertir les données de message d'application en la représentation requise par l'application de réception. La conversion des données de message d'application est facultative et requiert que l'option GMCONV soit spécifiée dans l'appel MQGET.

Les aspects suivants de la conversion des données sont décrits:

- Traitement effectué par le gestionnaire de files d'attente en réponse à l'option GMCONV ; voir [«Traitement de la conversion sous IBM i»](#), à la page 1509.
- Conventions de traitement utilisées par le gestionnaire de files d'attente lors du traitement d'un format intégré ; ces conventions sont également recommandées pour les exits écrits par l'utilisateur. Voir [«Conventions de traitement sous IBM i»](#), à la page 1510.
- Remarques spéciales sur la conversion des messages de rapport ; voir [«Conversion des messages de rapport sur IBM i»](#), à la page 1514.
- Les paramètres transmis à l'exit de conversion de données ; voir [«MQCONVX \(exit de conversion de données\) sous IBM i»](#), à la page 1525.
- Appel pouvant être utilisé à partir de l'exit pour convertir des données de type caractères entre différentes représentations ; voir [«MQXCNVC \(conversion de caractères\) sous IBM i»](#), à la page 1520.
- Paramètre de structure de données spécifique à l'exit ; voir [«MQDXP \(paramètre d'exit de conversion de données\) sous IBM i»](#), à la page 1515.

## IBM i Traitement de la conversion sous IBM i

Ces informations décrivent le traitement effectué par le gestionnaire de files d'attente en réponse à l'option GMCONV.

Le gestionnaire de files d'attente effectue les actions suivantes si l'option GMCONV est spécifiée sur l'appel MQGET et qu'un message doit être renvoyé à l'application:

1. Si une ou plusieurs des conditions suivantes sont remplies, aucune conversion n'est nécessaire:
  - Les données de message se trouvent déjà dans le jeu de caractères et le codage requis par l'application émettant l'appel MQGET. L'application doit définir les zones *MDCSI* et *MDENC* dans le paramètre **MSGDSC** de l'appel MQGET sur les valeurs requises, avant d'émettre l'appel.
  - La longueur des données de message est égale à zéro.
  - La longueur du paramètre **BUFFER** de l'appel MQGET est égale à zéro.

Dans ces cas, le message est renvoyé sans conversion à l'application émettant l'appel MQGET ; les valeurs *MDCSI* et *MDENC* du paramètre **MSGDSC** sont définies sur les valeurs des informations de contrôle du message et l'appel se termine avec l'une des combinaisons suivantes de code achèvement et de code anomalie:

### Code de fin d'exécution

#### Code raison

#### CCOK

RCNONE

#### CCWARN

RC2079

#### CCWARN

RC2080

Les étapes suivantes sont effectuées uniquement si le jeu de caractères ou le codage des données de message diffère de la valeur correspondante dans le paramètre **MSGDSC** et que des données doivent être converties:

1. Si la zone *MDFMT* des informations de contrôle du message a la valeur *FMNONE*, le message est renvoyé non converti, avec le code achèvement **CCWARN** et le code anomalie **RC2110**.

Dans tous les autres cas, le traitement de la conversion se poursuit.

2. Le message est supprimé de la file d'attente et placé dans une mémoire tampon temporaire de même taille que le paramètre **BUFFER** . Pour les opérations de navigation, le message est copié dans la mémoire tampon temporaire au lieu d'être supprimé de la file d'attente.
3. Si le message doit être tronqué pour tenir dans la mémoire tampon, les opérations suivantes sont effectuées:
  - Si l'option GMATM n'a pas été spécifiée, le message est renvoyé non converti, avec le code achèvement CCWARN et le code anomalie RC2080.
  - Si l'option GMATM a été spécifiée, le code achèvement est défini sur CCWARN, le code anomalie est défini sur RC2079 et le traitement de la conversion se poursuit.
4. Si le message peut être placé dans la mémoire tampon sans troncature, ou si l'option GMATM a été spécifiée, procédez comme suit:
  - Si le format est un format intégré, la mémoire tampon est transmise au service de conversion de données du gestionnaire de files d'attente.
  - Si le format n'est pas un format intégré, la mémoire tampon est transmise à un exit écrit par l'utilisateur ayant le même nom que le format. Si l'exit est introuvable, le message est renvoyé non converti, avec le code achèvement CCWARN et le code anomalie RC2110.

Si aucune erreur ne se produit, la sortie du service de conversion de données ou de l'exit écrit par l'utilisateur est le message converti, ainsi que le code achèvement et le code raison à renvoyer à l'application émettant l'appel MQGET.
5. Si la conversion aboutit, le gestionnaire de files d'attente renvoie le message converti à l'application. Dans ce cas, le code achèvement et le code anomalie renvoyés par l'appel MQGET sont généralement l'une des combinaisons suivantes:

**Code de fin d'exécution**  
**Code raison**

**CCOK**  
 RCNONE

**CCWARN**  
 RC2079

Toutefois, si la conversion est effectuée par un exit écrit par l'utilisateur, d'autres codes raison peuvent être renvoyés, même lorsque la conversion aboutit.

Si la conversion échoue (pour quelque raison que ce soit), le gestionnaire de files d'attente renvoie le message non converti à l'application, avec les zones *MDCSI* et *MDENC* du paramètre **MSGDSC** définies sur les valeurs des informations de contrôle du message et avec le code achèvement CCWARN.

## IBM i Conventions de traitement sous IBM i

Lors de la conversion d'un format intégré, le gestionnaire de files d'attente respecte les conventions de traitement décrites dans cette rubrique.

Envisagez d'appliquer ces conventions aux exits écrits par l'utilisateur, bien que cela ne soit pas appliqué par le gestionnaire de files d'attente. Les formats intégrés convertis par le gestionnaire de files d'attente sont les suivants:

- FMADMN
- FMMDE
- FMCICS
- FMPCF
- FMCMD1
- FMRMH
- FMCMD2
- FMRFH

- FMDLH
- FMRFH2
- FMDH
- FSTR
- FMEVNT
- FMTM
- FMIMS
- FMXQH
- FMIMVS

1. Si le message se développe lors de la conversion et dépasse la taille du paramètre **BUFFER** , procédez comme suit:

- Si l'option GMATM n'a pas été spécifiée, le message est renvoyé non converti, avec le code achèvement CCWARN et le code anomalie RC2120.
- Si l'option GMATM a été spécifiée, le message est tronqué, le code achèvement est défini sur CCWARN, le code anomalie est défini sur RC2079 et le traitement de la conversion se poursuit.

2. En cas de troncature (avant ou pendant la conversion), il est possible que le nombre d'octets valides renvoyés dans le paramètre **BUFFER** soit *inférieur* à la longueur de la mémoire tampon.

Cela peut se produire, par exemple, si un entier de 4 octets ou un caractère DBCS chevauche la fin de la mémoire tampon. L'élément incomplet des informations n'est pas converti, de sorte que les octets du message renvoyé ne contiennent pas d'informations valides. Cela peut également se produire si un message tronqué avant la conversion est réduit lors de la conversion.

Si le nombre d'octets valides renvoyés est inférieur à la longueur de la mémoire tampon, les octets inutilisés à la fin de la mémoire tampon sont définis sur null.

3. Si un tableau ou une chaîne chevauche la fin de la mémoire tampon, la plus grande partie possible des données est convertie ; seul l'élément de tableau ou le caractère DBCS qui est incomplet est converti-les éléments de tableau ou les caractères précédents sont convertis.

4. Si une troncature est effectuée (avant ou pendant la conversion), la longueur renvoyée pour le paramètre **DATLEN** est la longueur du message *non converti* avant la troncature.

5. Lorsque des chaînes sont converties entre des jeux de caractères codés sur un octet (SBCS), des jeux de caractères codés sur deux octets (DBCS) ou des jeux de caractères codés sur plusieurs octets (MBCS), les chaînes peuvent se développer ou se contracter.

- Dans les formats PCF FMADMN, FMEVNT et FMPCF, les chaînes des structures MQCFST et MQCFSL se développent ou se contractent selon les besoins pour s'adapter à la chaîne après la conversion.

Pour la structure de liste de chaînes MQCFSL, les chaînes de la liste peuvent se développer ou se contracter différemment. Dans ce cas, le gestionnaire de files d'attente remplit les chaînes plus courtes avec des blancs pour qu'elles aient la même longueur que la chaîne la plus longue après la conversion.

- Dans le format FMRMH, les chaînes adressées par les zones RMSEO, RMSNO, RMDEO et RMDNO se développent ou se contractent selon les besoins pour s'adapter aux chaînes après la conversion.

- Dans le format FMRFH, la zone RFNVS se développe ou se contracte selon les besoins pour s'adapter aux paires nom-valeur après la conversion.

- Dans les structures avec des tailles de champ fixes, le gestionnaire de files d'attente permet aux chaînes de se développer ou de se contracter dans leurs champs fixes, si aucune information significative n'est perdue. A cet égard, les blancs de fin et les caractères qui suivent le premier caractère nul de la zone sont considérés comme non significatifs.

- Si la chaîne se développe, mais que seuls des caractères non significatifs doivent être supprimés pour accueillir la chaîne convertie dans la zone, la conversion aboutit et l'appel se termine avec CCOK et le code raison RCNONE (en supposant qu'il n'y ait pas d'autres erreurs).

- Si la chaîne se développe, mais que la chaîne convertie nécessite que des caractères significatifs soient supprimés pour tenir dans la zone, le message est renvoyé non converti et l'appel se termine avec CCWARN et le code anomalie RC2190.

**Remarque :** Le code anomalie RC2190 indique dans ce cas si l'option GMATM a été spécifiée.

- Si la chaîne se contracte, le gestionnaire de files d'attente complète la chaîne avec des blancs à la longueur de la zone.

6. Pour les messages constitués d'une ou de plusieurs structures d'en-tête IBM MQ suivies de données utilisateur, il est possible de convertir une ou plusieurs structures d'en-tête, alors que le reste du message ne l'est pas. Toutefois, à deux exceptions près, les zones MDCSI et MDENC de chaque structure d'en-tête indiquent toujours correctement le jeu de caractères et le codage des données qui suivent la structure d'en-tête.

Les deux exceptions sont les structures MQCIH et MQIIH, où les valeurs des zones MDCSI et MDENC de ces structures ne sont pas significatives. Pour ces structures, les données qui suivent la structure se trouvent dans le même jeu de caractères et le même codage que la structure MQCIH ou MQIIH elle-même.

7. Si les zones MDCSI ou MDENC dans les informations de contrôle du message en cours d'extraction ou dans le paramètre **MSGDSC** indiquent des valeurs non définies ou non prises en charge, le gestionnaire de files d'attente peut ignorer l'erreur si la valeur non définie ou non prise en charge n'a pas besoin d'être utilisée pour convertir le message.

Par exemple, si la zone MDENC du message spécifie un codage à virgule flottante non pris en charge, mais que le message contient uniquement des données entières, ou contient des données à virgule flottante qui ne nécessitent pas de conversion (car les codages à virgule flottante source et cible sont identiques), l'erreur peut ou non être diagnostiquée.

Si l'erreur est diagnostiquée, le message est renvoyé non converti, avec le code achèvement CCWARN et l'un des codes anomalie RC2111, RC2112, RC2113, RC2114 ou RC2115, RC2116, RC2117, RC2118 (selon le cas) ; les zones MDCSI et MDENC du paramètre **MSGDSC** sont définies sur les valeurs des informations de contrôle du message.

Si l'erreur n'est pas diagnostiquée et que la conversion aboutit, les valeurs renvoyées dans les zones MDCSI et MDENC du paramètre **MSGDSC** sont celles spécifiées par l'application qui émet l'appel MQGET.

8. Dans tous les cas, si le message est renvoyé à l'application non convertie, le code achèvement est défini sur CCWARN et les zones MDCSI et MDENC du paramètre **MSGDSC** sont définies sur les valeurs appropriées aux données non converties. Cette opération est également effectuée pour FMNONE.

Le paramètre **REASON** est défini sur un code qui indique la raison pour laquelle la conversion n'a pas pu être effectuée, sauf si le message a également dû être tronqué ; les codes raison liés à la troncature sont prioritaires sur les codes raison liés à la conversion. (Pour déterminer si un message tronqué a été converti, vérifiez les valeurs renvoyées dans les zones MDCSI et MDENC du paramètre **MSGDSC**.)

Lorsqu'une erreur est diagnostiquée, un code anomalie spécifique est renvoyé ou le code anomalie général est RC2119. Le code anomalie renvoyé dépend des capacités de diagnostic du service de conversion de données sous-jacent.

9. Si le code achèvement CCWARN est renvoyé et que plusieurs codes raison sont pertinents, l'ordre de priorité est le suivant:
  - a. La raison suivante prévaut sur toutes les autres:
    - RC2079
  - b. La raison suivante de la priorité est la suivante:
    - RC2110
  - c. L'ordre de priorité dans les codes anomalie restants n'est pas défini.

10. A la fin de l'appel MQGET:



- Le code anomalie suivant indique que la conversion du message a abouti:
  - RCNONE
- Le code anomalie suivant indique que le message *peut* avoir été converti (vérifiez les zones MDCSI et MDENC dans le paramètre **MSGDSC** pour le savoir):
  - RC2079
- Tous les autres codes anomalie indiquent que le message n'a pas été converti.

Le traitement suivant est spécifique aux formats intégrés ; il ne s'applique pas aux formats définis par l'utilisateur:

1. Sauf pour les formats suivants:

- FMADMN
- FMEVNT
- FMIMVS
- FMPCF
- FSTR

aucun des formats intégrés ne peut être converti depuis ou vers des jeux de caractères qui ne comportent pas de caractères SBCS pour les caractères admis dans les noms de file d'attente. Si une tentative est effectuée pour effectuer une telle conversion, le message est renvoyé non converti, avec le code achèvement CCWARN et le code anomalie RC2111 ou RC2115, selon le cas.

Le jeu de caractères Unicode UTF-16 est un exemple de jeu de caractères qui ne comporte pas de caractères SBCS pour les caractères valides dans les noms de file d'attente.

2. Si les données de message pour un format intégré sont tronquées, les zones du message qui contiennent des longueurs de chaînes ou des nombres d'éléments ou de structures ne sont pas ajustées pour refléter la longueur des données renvoyées à l'application ; les valeurs renvoyées pour ces zones dans les données de message sont les valeurs applicables au message avant troncature.

Lors du traitement de messages tels qu'un message FMADMN tronqué, veillez à ce que l'application ne tente pas d'accéder à des données au-delà de la fin des données renvoyées.

3. Si le nom de format est FMDLH, les données de message commencent par une structure MQDLH, qui peut être suivie de zéro ou plusieurs octets de données de message d'application. Le format, le jeu de caractères et le codage des données de message d'application sont définis par les zones DLFMT, DLCSi et DLENC de la structure MQDLH au début du message. Etant donné que la structure MQDLH et les données de message d'application peuvent avoir des jeux de caractères et des codages différents, il est possible que l'une ou l'autre, ou les deux, de la structure MQDLH et des données de message d'application nécessitent une conversion.

Le gestionnaire de files d'attente convertit d'abord la structure MQDLH, si nécessaire. Si la conversion aboutit ou si la structure MQDLH ne requiert pas de conversion, le gestionnaire de files d'attente vérifie les zones DLCSi et DLENC de la structure MQDLH pour déterminer si la conversion des données de message d'application est requise. Si une conversion est requise, le gestionnaire de files d'attente appelle l'exit écrit par l'utilisateur avec le nom donné par la zone DLFMT dans la structure MQDLH ou effectue la conversion elle-même (si DLFMT est le nom d'un format intégré).

Si l'appel MQGET renvoie le code achèvement CCWARN et que le code anomalie est l'un de ceux indiquant que la conversion a échoué, l'un des cas suivants s'applique:

- La structure MQDLH n'a pas pu être convertie. Dans ce cas, les données de message d'application n'auront pas non plus été converties.
- La structure MQDLH a été convertie, mais les données de message d'application ne l'ont pas été.

L'application peut examiner les valeurs renvoyées dans les zones MDCSI et MDENC du paramètre **MSGDSC**, ainsi que celles de la structure MQDLH, afin de déterminer laquelle des précédentes s'applique.

4. Si le nom de format est FMXQH, les données de message commencent par une structure MQXQH, qui peut être suivie de zéro ou plusieurs octets de données supplémentaires. Ces données supplémentaires sont généralement les données de message d'application (qui peuvent être de longueur nulle), mais il peut également y avoir une ou plusieurs autres structures d'en-tête IBM MQ, au début des données supplémentaires.

La structure MQXQH doit être dans le jeu de caractères et le codage du gestionnaire de files d'attente. Le format, le jeu de caractères et le codage des données qui suivent la structure MQXQH sont fournis par les zones MDFMT, MDCSI et MDENC de la structure MQMD contenue dans MQXQH. Pour chaque structure d'en-tête IBM MQ suivante présente, les zones MDFMT, MDCSI et MDENC de la structure décrivent les données qui suivent cette structure ; ces données sont soit une autre structure d'en-tête IBM MQ, soit les données de message d'application.

Si l'option GMCONV est spécifiée pour un message FMXQH, les données de message d'application et certaines des structures d'en-tête MQ sont converties, mais pas les données de la structure MQXQH. En cas de retour de l'appel MQGET, par conséquent :

- Les valeurs des zones MDFMT, MDCSI et MDENC dans le paramètre **MSGDSC** décrivent les données dans la structure MQXQH et non les données de message d'application ; les valeurs ne seront donc pas identiques à celles spécifiées par l'application qui a émis l'appel MQGET.

En conséquence, une application qui extrait à plusieurs reprises des messages d'une file d'attente de transmission avec l'option GMCONV spécifiée doit réinitialiser les zones MDCSI et MDENC du paramètre **MSGDSC** aux valeurs nécessaires pour les données de message d'application, avant chaque appel MQGET.

- Les valeurs des zones MDFMT, MDCSI et MDENC de la dernière structure d'en-tête MQ présente décrivent les données du message d'application. Si aucune autre structure d'en-tête IBM MQ n'est présente, les données de message d'application sont décrites par ces zones dans la structure MQMD au sein de la structure MQXQH. Si la conversion aboutit, les valeurs seront identiques à celles spécifiées dans le paramètre **MSGDSC** par l'application qui a émis l'appel MQGET.

Si le message est un message de liste de distribution, la structure MQXQH est suivie d'une structure MQDH (plus ses tableaux d'enregistrements MQOR et MQPMR), qui à son tour peut être suivie de zéro ou plusieurs autres structures d'en-tête IBM MQ et de zéro ou plusieurs octets de données de message d'application. À l'instar de la structure MQXQH, la structure MQDH doit être dans le jeu de caractères et le codage du gestionnaire de files d'attente et elle n'est pas convertie lors de l'appel MQGET, même si l'option GMCONV est spécifiée.

Le traitement des structures MQXQH et MQDH décrites précédemment est principalement destiné à être utilisé par les agents MCA lorsqu'ils extraient des messages des files d'attente de transmission.



## Conversion des messages de rapport sur IBM i

Un message de rapport peut contenir des quantités variables de données de message d'application, en fonction des options de rapport spécifiées par l'expéditeur du message d'origine.

En particulier, un message de rapport peut contenir :

1. Aucune donnée de message d'application
2. Certaines des données de message d'application du message d'origine

Cela se produit lorsque l'expéditeur du message d'origine spécifie RO\* D et que le message dépasse 100 octets.

3. Toutes les données de message d'application du message d'origine

Cela se produit lorsque l'expéditeur du message d'origine indique RO\* F ou RO\* D et que le message est de 100 octets ou moins.

Lorsque le gestionnaire de files d'attente ou l'agent MCA génère un message de rapport, il copie le nom de format du message d'origine dans la zone *MDFMT* des informations de contrôle du message de rapport. Le nom de format dans le message de rapport peut donc impliquer une longueur de données différente de la longueur présente dans le message de rapport (cas 1 et 2 décrits précédemment).

Si l'option GMCONV est spécifiée lors de l'extraction du message de rapport:

- Pour le cas 1 décrit précédemment, l'exit de conversion de données ne sera pas appelé (car le message de rapport ne contient pas de données).
- Pour le cas 3 décrit précédemment, le nom de format implique correctement la longueur des données de message.
- Mais pour le cas 2 décrit précédemment, l'exit de conversion de données sera appelé pour convertir un message qui est *plus court* que la longueur impliquée par le nom de format.

En outre, le code raison transmis à l'exit est généralement RCNONE (c'est-à-dire que le code raison n'indique pas que le message a été tronqué). Cela se produit car les données de message ont été tronquées par l'*émetteur* du message de rapport et non par le gestionnaire de files d'attente du récepteur en réponse à l'appel MQGET.

En raison de ces possibilités, l'exit de conversion de données ne doit pas utiliser le nom de format pour déduire la longueur des données qui lui sont transmises ; l'exit doit plutôt vérifier la longueur des données fournies et être prêt à convertir moins de données que la longueur impliquée par le nom de format. Si les données peuvent être converties avec succès, le code achèvement CCOK et le code raison RCNONE doivent être renvoyés par l'exit. La longueur des données de message à convertir est transmise à l'exit en tant que paramètre **INLEN**.

## Interface de programmation sensible au produit

Si un message de rapport contient des informations sur une activité qui a eu lieu, il s'agit d'un rapport d'activité. Voici des exemples d'activités:

- un agent MCA envoyant un message à partir d'une file d'attente vers un canal
- un agent MCA recevant un message d'un canal et le plaçant dans une file d'attente
- une file d'attente de rebut MCA mettant en file d'attente un message non distribuable
- un agent MCA obtenant un message d'une file d'attente et le supprimant
- un gestionnaire de rebut remplaçant un message dans une file d'attente
- le serveur de commandes traitant une demande PCF-un courtier traitant une demande de publication
- une application utilisateur obtenant un message d'une file d'attente-une application utilisateur parcourant un message d'une file d'attente

Toute application, y compris le gestionnaire de files d'attente, peut ajouter une partie des données de message au rapport d'activité après l'en-tête du rapport. La quantité de données à fournir en cas d'envoi n'est pas fixe et est déterminée par l'application. Les informations renvoyées doivent être utiles à l'application qui traite le rapport d'activité. Les rapports d'activité du gestionnaire de files d'attente renvoient avec eux toutes les structures d'en-tête IBM MQ standard (commençant par'MQH') contenues dans le message d'origine. Cela inclut, par exemple, tous les en-têtes MQRFH2 qui ont été inclus dans le message d'origine. Le gestionnaire de files d'attente renvoie également un en-tête MQCFH trouvé, mais pas les paramètres PCF qui lui sont associés. Cela donne aux applications de surveillance une idée de la nature du message.

IBM i

## MQDXP (paramètre d'exit de conversion de données) sous IBM i

Bloc de paramètres d'exit de conversion de données.

### Présentation

**Objectif:** La structure MQDXP est un paramètre que le gestionnaire de files d'attente transmet à l'exit de conversion de données lorsque l'exit est appelé pour convertir les données de message dans le cadre du traitement de l'appel MQGET. Pour plus de détails sur l'exit de conversion de données, voir la description de l'appel MQCONVX.

**Jeu de caractères et codage:** les données de type caractère dans MQDXP se trouvent dans le jeu de caractères du gestionnaire de files d'attente local ; elles sont fournies par l'attribut de gestionnaire de

files d'attente **CodedCharSetId** . Les données numériques dans MQDXP sont dans le codage de la machine native ; elles sont fournies par ENNAT.

**Utilisation:** Seules les zones *DXLEN*, *DXCC*, *DXREA* et *DXRES* de MQDXP peuvent être modifiées par l'exit ; les modifications apportées aux autres zones sont ignorées. Toutefois, la zone *DXLEN* ne peut pas être modifiée si le message en cours de conversion est un segment qui ne contient qu'une partie d'un message logique.

Lorsque le contrôle revient au gestionnaire de files d'attente à partir de l'exit, le gestionnaire de files d'attente vérifie les valeurs renvoyées dans MQDXP. Si les valeurs renvoyées ne sont pas valides, le gestionnaire de files d'attente poursuit le traitement comme si l'exit avait renvoyé XRFAIL dans *DXRES* ; toutefois, le gestionnaire de files d'attente ignore les valeurs des zones *DXCC* et *DXREA* renvoyées par l'exit dans ce cas, et utilise à la place les valeurs que ces zones avaient dans les *entrées* de l'exit. Les valeurs suivantes dans MQDXP entraînent ce traitement:

- Zone *DXRES* non XROK et non XRFAIL
- Zone *DXCC* non CCOK et non CCWARN
- Zone *DXLEN* inférieure à zéro ou zone *DXLEN* modifiée lorsque le message en cours de conversion est un segment qui ne contient qu'une partie d'un message logique.
- «Zones», à la page 1516
- «Déclaration RPG (copie du fichier CMQDXPH)», à la page 1520

## Zones

La structure MQDXP contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

### **DXAOP (entier signé à 10 chiffres)**

Options d'application.

Il s'agit d'une copie de la zone *GMOPT* de la structure MQGMO spécifiée par l'application émettant l'appel MQGET. L'exit peut avoir besoin de les examiner pour déterminer si l'option GMATM a été spécifiée.

Il s'agit d'une zone d'entrée de l'exit.

### **DXCC (entier signé à 10 chiffres)**

Code achèvement.

Lorsque l'exit est appelé, il contient le code achèvement qui sera renvoyé à l'application qui a émis l'appel MQGET, si l'exit choisit de ne rien faire. Il s'agit toujours de CCWARN, car soit le message a été tronqué, soit le message doit être converti, ce qui n'a pas encore été fait.

Dans la sortie de l'exit, cette zone contient le code achèvement à renvoyer à l'application dans le paramètre **CMPCOD** de l'appel MQGET ; seuls CCOK et CCWARN sont valides. Consultez la description de la zone *DXREA* pour obtenir des suggestions sur la manière dont l'exit doit définir cette zone sur la sortie.

Il s'agit d'une zone d'entrée-sortie pour l'exit.

### **DXCSI (entier signé à 10 chiffres)**

Jeu de caractères requis par l'application.

Il s'agit de l'identificateur de jeu de caractères codés du jeu de caractères requis par l'application émettant l'appel MQGET ; voir la zone *MDCSI* dans la structure MQMD pour plus de détails. Si l'application spécifie la valeur spéciale CSQM dans l'appel MQGET, le gestionnaire de files d'attente la remplace par l'identificateur de jeu de caractères réel du jeu de caractères utilisé par le gestionnaire de files d'attente, avant d'appeler l'exit.

Si la conversion aboutit, l'exit doit le copier dans la zone *MDCSI* du descripteur de message.

Il s'agit d'une zone d'entrée de l'exit.

### **DXENC (entier signé à 10 chiffres)**

Codage numérique requis par l'application.

Il s'agit du codage numérique requis par l'application qui émet l'appel MQGET ; voir la zone *MDENC* dans la structure MQMD pour plus de détails.

Si la conversion aboutit, l'exit doit le copier dans la zone *MDENC* du descripteur de message.

Il s'agit d'une zone d'entrée de l'exit.

### **DXHCN (entier signé à 10 chiffres)**

Descripteur de connexion.

Il s'agit d'un descripteur de connexion qui peut être utilisé sur l'appel MQXCNCV. Ce descripteur n'est pas nécessairement le même que celui spécifié par l'application qui a émis l'appel MQGET.

### **DXLEN (entier signé à 10 chiffres)**

Longueur en octets des données de message.

Lorsque l'exit est appelé, cette zone contient la longueur d'origine des données de message d'application. Si le message a été tronqué pour tenir dans la mémoire tampon fournie par l'application, la taille du message fourni à l'exit sera *inférieure* à la valeur de *DXLEN*. La taille du message fourni à l'exit est toujours indiquée par le paramètre **INLEN** de l'exit, quelle que soit la troncature qui a pu se produire.

La troncature est indiquée par la zone *DXREA* dont la valeur est RC2079 en entrée de l'exit.

La plupart des conversions n'ont pas besoin de modifier cette longueur, mais un exit peut le faire si nécessaire ; la valeur définie par l'exit est renvoyée à l'application dans le paramètre **DATLEN** de l'appel MQGET. Toutefois, cette longueur ne peut pas être modifiée si le message en cours de conversion est un segment qui ne contient qu'une partie d'un message logique. En effet, si vous modifiez la longueur, les décalages des segments ultérieurs dans le message logique seront incorrects.

Notez que, si l'exit souhaite modifier la longueur des données, sachez que le gestionnaire de files d'attente a déjà décidé si les données de message s'insèrent dans la mémoire tampon de l'application, en fonction de la longueur des données *non converties*. Cette décision détermine si le message est supprimé de la file d'attente (ou si le curseur de navigation a été déplacé pour une demande de navigation) et s'il n'est pas affecté par une modification de la longueur des données provoquée par la conversion. Pour cette raison, il est recommandé que les exits de conversion n'entraînent pas de modification de la longueur des données de message d'application.

Si la conversion de caractères implique un changement de longueur, une chaîne peut être convertie en une autre chaîne de même longueur en octets, en tronquant les blancs de fin ou en ajoutant des blancs si nécessaire.

L'exit n'est pas appelé si le message ne contient pas de données de message d'application ; par conséquent, *DXLEN* est toujours supérieur à zéro.

Il s'agit d'une zone d'entrée-sortie pour l'exit.

### **DXREA (entier signé à 10 chiffres)**

Code anomalie qualifiant *DXCC*.

Lorsque l'exit est appelé, il contient le code anomalie qui sera renvoyé à l'application qui a émis l'appel MQGET, si l'exit choisit de ne rien faire. Parmi les valeurs possibles figurent RC2079, qui indique que le message a été tronqué pour tenir dans la mémoire tampon fournie par l'application, et RC2119, qui indique que le message requiert une conversion mais que cette opération n'a pas encore été effectuée.

Dans la sortie de l'exit, cette zone contient la raison du renvoi à l'application dans le paramètre **REASON** de l'appel MQGET ; les éléments suivants sont recommandés:

- Si *DXREA* a la valeur *RC2079* en entrée de l'exit, les zones *DXREA* et *DXCC* ne doivent pas être modifiées, que la conversion aboutisse ou échoue.

(Si la zone *DXCC* n'est pas *CCOK*, l'application qui extrait le message peut identifier un échec de conversion en comparant les valeurs *MDENC* et *MDCSI* renvoyées dans le descripteur de message avec les valeurs demandées ; en revanche, l'application ne peut pas distinguer un message tronqué d'un message qui vient d'être adapté à la mémoire tampon. Pour cette raison, *RC2079* doit être renvoyé de préférence à l'une des raisons qui indiquent l'échec de la conversion.)

- Si *DXREA* avait une autre valeur en entrée de l'exit:
  - Si la conversion aboutit, *DXCC* doit être défini sur *CCOK* et *DXREA* sur *RCNONE*.
  - Si la conversion échoue ou si le message se développe et doit être tronqué pour tenir dans la mémoire tampon, *DXCC* doit être défini sur *CCWARN* (ou laissé inchangé), et *DXREA* défini sur l'une des valeurs *i* de la liste suivante, pour indiquer la nature de l'échec.

Notez que si le message après conversion est trop volumineux pour la mémoire tampon, il ne doit être tronqué que si l'application qui a émis l'appel *MQGET* a spécifié l'option *GMATM*:

- S'il a spécifié cette option, le motif *RC2079* doit être renvoyé.
- S'il n'a pas spécifié cette option, le message doit être renvoyé non converti, avec le code anomalie *RC2120*.

Les codes raison de la liste suivante sont recommandés pour être utilisés par l'exit afin d'indiquer la raison pour laquelle la conversion a échoué, mais l'exit peut renvoyer d'autres valeurs à partir de l'ensemble de codes *RC\** s'il le juge approprié. De plus, la plage de valeurs *RC0900* à *RC0999* est allouée à l'exit pour indiquer les conditions dans lesquelles l'exit souhaite communiquer avec l'application qui émet l'appel *MQGET*.

**Remarque :** Si le message ne peut pas être converti correctement, l'exit doit renvoyer *XRFAIL* dans la zone *DXRES* afin que le gestionnaire de files d'attente renvoie le message non converti. Cela est vrai quel que soit le code anomalie renvoyé dans la zone *DXREA* .

#### **RC0900**

(900, X'384') Valeur la plus faible pour le code anomalie défini par l'application.

#### **RC0999**

(999, X'3E7') Valeur la plus élevée pour le code anomalie défini par l'application.

#### **RC2120**

(2120, X'848') Données converties trop volumineuses pour la mémoire tampon.

#### **RC2119**

(2119, X'847') Données de message non converties.

#### **RC2111**

(2111, X'83F') Identificateur de jeu de caractères codés source non valide.

#### **RC2113**

(2113, X'841') Codage décimal condensé dans le message non reconnu.

#### **RC2114**

(2114, X'842') Codage à virgule flottante dans le message non reconnu.

#### **RC2112**

(2112, X'840') Codage d'entier source non reconnu.

#### **RC2115**

(2115, X'843') Identificateur de jeu de caractères codés cible non valide.

#### **RC2117**

(2117, X'845') Codage décimal condensé spécifié par le récepteur non reconnu.

#### **RC2118**

(2118, X'846') Codage à virgule flottante spécifié par le récepteur non reconnu.

#### **RC2116**

(2116, X'844') Codage d'entier cible non reconnu.

**RC2079**

(2079, X'81F') Message tronqué renvoyé (traitement terminé).

Il s'agit d'une zone d'entrée-sortie pour l'exit.

**DXRES (entier signé à 10 chiffres)**

Réponse de l'exit.

Ce paramètre est défini par l'exit pour indiquer la réussite ou non de la conversion. Il doit s'agir de l'une des valeurs suivantes:

**XR0K**

La conversion a abouti.

Si l'exit spécifie cette valeur, le gestionnaire de files d'attente renvoie la valeur suivante à l'application qui a émis l'appel MQGET:

- Valeur de la zone *DXCC* dans la sortie de l'exit
- Valeur de la zone *DXREA* dans la sortie de l'exit
- Valeur de la zone *DXLEN* dans la sortie de l'exit
- Contenu de la mémoire tampon de sortie de l'exit *OUTBUF*. Le nombre d'octets renvoyés est le plus petit du paramètre **OUTLEN** de l'exit et la valeur de la zone *DXLEN* dans la sortie de l'exit

Si les zones *MDENC* et *MDCSI* du paramètre de descripteur de message de l'exit sont *les deux* inchangées, le gestionnaire de files d'attente renvoie:

- Valeur des zones *MDENC* et *MDCSI* dans la structure MQDXP sur l' *entrée* de l'exit

Si l'une ou les deux zones *MDENC* et *MDCSI* du paramètre de descripteur de message de l'exit ont été modifiées, le gestionnaire de files d'attente renvoie:

- Valeur des zones *MDENC* et *MDCSI* dans le paramètre de descripteur de message de l'exit sur la sortie de l'exit
- 

**XRFAIL**

La conversion a échoué.

Si l'exit spécifie cette valeur, le gestionnaire de files d'attente renvoie la valeur suivante à l'application qui a émis l'appel MQGET:

- Valeur de la zone *DXCC* dans la sortie de l'exit
- Valeur de la zone *DXREA* dans la sortie de l'exit
- Valeur de la zone *DXLEN* dans l' *entrée* de l'exit
- Contenu de la mémoire tampon d'entrée de l'exit *INBUF*. Le nombre d'octets renvoyés est indiqué par le paramètre **INLEN**

Si l'exit a modifié *INBUF*, les résultats ne sont pas définis.

*DXRES* est une zone de sortie de l'exit.

**DXSID (chaîne de caractères de 4 octets)**

Identificateur de structure.

La valeur doit être:

**DXSIDV**

Identificateur de la structure des paramètres d'exit de conversion de données.

Il s'agit d'une zone d'entrée de l'exit.

**DXVER (entier signé à 10 chiffres)**

Numéro de version de la structure.

La valeur doit être:

#### **DXVER1**

Numéro de version de la structure des paramètres d'exit de conversion de données.

La constante suivante indique le numéro de version de la version en cours:

#### **DXVERC**

Version actuelle de la structure des paramètres d'exit de conversion de données.

**Remarque :** Lorsqu'une nouvelle version de cette structure est introduite, la présentation de la partie existante n'est pas modifiée. L'exit doit donc vérifier que la zone *DXVER* est égale ou supérieure à la version la plus basse qui contient les zones que l'exit doit utiliser.

Il s'agit d'une zone d'entrée de l'exit.

#### **DXXOP (entier signé à 10 chiffres)**

Réservé.

Il s'agit d'une zone réservée ; sa valeur est 0.

### **Déclaration RPG (copie du fichier CMQDXPH)**

```
D* .1.....2.....3.....4.....5.....6.....7..
D* MQDXP Structure
D*
D* Structure identifier
D DXSID 1 4
D* Structure version number
D DXVER 5 8I 0
D* Reserved
D DXXOP 9 12I 0
D* Application options
D DXAOP 13 16I 0
D* Numeric encoding required by application
D DXENC 17 20I 0
D* Character set required by application
D DXCSI 21 24I 0
D* Length in bytes of message data
D DXLEN 25 28I 0
D* Completion code
D DXCC 29 32I 0
D* Reason code qualifying DXCC
D DXREA 33 36I 0
D* Response from exit
D DXRES 37 40I 0
D* Connection handle
D DXHCN 41 44I 0
```

**IBM i**

### **MQXCNCV (conversion de caractères) sous IBM i**

L'appel MQXCNCV convertit les caractères d'un jeu de caractères à un autre.

Cet appel fait partie de l'interface de conversion de données IBM MQ (DCI), qui est l'une des interfaces de l'infrastructure IBM MQ . Remarque: Cet appel peut être utilisé uniquement à partir d'un exit de conversion de données.

- «Syntaxe», à la page [1521](#)
- «Paramètres», à la page [1521](#)
- «Appel RPG (ILE)», à la page [1525](#)



## Syntaxe

**HCONN, OPTS, SRCCSI, SRCLEN, SRCBUF, TGTCSE, TGTLEN, MQXCNCV**  
(TGTBUF, DATLEN, CMPCOD, REASON)

## Paramètres

L'appel MQXCNCV comporte les paramètres suivants:

### **HCONN (entier signé à 10 chiffres)-entrée**

Descripteur de connexion.

Ce descripteur représente la connexion au gestionnaire de files d'attente. Il doit normalement s'agir du descripteur transmis à l'exit de conversion de données dans la zone DXHCN de la structure MQDXP ; ce descripteur n'est pas nécessairement le même que celui spécifié par l'application qui a émis l'appel MQGET.

Sous IBM i, la valeur spéciale suivante peut être spécifiée pour HCONN:

#### **HCDEFH**

Descripteur de connexion par défaut.

### **OPTS (entier signé à 10 chiffres)-entrée**

Options qui contrôlent l'action de MQXCNCV.

Aucune ou plusieurs des options décrites plus loin dans cette section peuvent être spécifiées. Si plusieurs valeurs sont requises, les valeurs peuvent être ajoutées (n'ajoutez pas la même constante plusieurs fois).

**Option de conversion par défaut:** L'option suivante contrôle l'utilisation de la conversion de caractères par défaut:

#### **DCCDEF**

Conversion par défaut.

Cette option indique que la conversion de caractères par défaut peut être utilisée si l'un des jeux de caractères indiqués dans l'appel, ou les deux, ne sont pas pris en charge. Cela permet au gestionnaire de files d'attente d'utiliser un jeu de caractères par défaut spécifié par l'installation qui se rapproche du jeu de caractères spécifié lors de la conversion de la chaîne.

**Remarque :** Le résultat de l'utilisation d'un jeu de caractères approximatif pour convertir la chaîne est que certains caractères peuvent être convertis de manière incorrecte. Cela peut être évité en utilisant dans la chaîne uniquement des caractères qui sont communs au jeu de caractères spécifié et au jeu de caractères par défaut.

Les jeux de caractères par défaut sont définis par une option de configuration lorsque le gestionnaire de files d'attente est installé ou redémarré.

Si DCCDEF n'est pas spécifié, le gestionnaire de files d'attente utilise uniquement les jeux de caractères spécifiés pour convertir la chaîne, et l'appel échoue si l'un ou les deux jeux de caractères ne sont pas pris en charge.

**Option de remplissage:** L'option suivante permet au gestionnaire de files d'attente de remplir la chaîne convertie avec des blancs ou de supprimer les caractères de fin non significatifs afin que la chaîne convertie corresponde à la mémoire tampon cible:

#### **DCCFIL**

Remplissage de la mémoire tampon cible.

Cette option demande que la conversion soit effectuée de telle sorte que la mémoire tampon cible soit complètement remplie:

- Si la chaîne se contracte lors de sa conversion, des blancs de fin sont ajoutés afin de remplir la mémoire tampon cible.

- Si la chaîne se développe lors de sa conversion, les caractères de fin qui ne sont pas significatifs sont supprimés pour que la chaîne convertie corresponde à la mémoire tampon cible. Si cela peut être effectué avec succès, l'appel se termine avec CCOK et le code anomalie RCNONE.

S'il y a trop peu de caractères de fin non significatifs, la partie de la chaîne qui convient est placée dans la mémoire tampon cible et l'appel se termine avec CCWARN et le code anomalie RC2120.

Les caractères non significatifs sont les suivants:

- Blancs de fin
- Caractères suivant le premier caractère null dans la chaîne (à l'exclusion du premier caractère null lui-même)
- Si la chaîne, TGTCSI et TGTLEN sont telles que la mémoire tampon cible ne peut pas être complètement définie avec des caractères valides, l'appel échoue avec CCFAIL et le code anomalie RC2144. Cela peut se produire lorsque TGTCSI est un jeu de caractères DBCS pur (tel que UTF-16), mais que TGTLEN spécifie une longueur qui est un nombre impair d'octets.
- TGTLEN peut être inférieur ou supérieur à SRCLen. Au retour de MQXCNCV, DATLEN a la même valeur que TGTLEN.

Si cette option n'est pas spécifiée:

- La chaîne est autorisée à se contracter ou à se développer dans la mémoire tampon cible selon les besoins. Les caractères de fin non significatifs ne sont pas ajoutés ou supprimés.

Si la chaîne convertie correspond à la mémoire tampon cible, l'appel se termine avec CCOK et le code anomalie RCNONE.

Si la chaîne convertie est trop grande pour la mémoire tampon cible, la quantité de la chaîne qui convient est placée dans la mémoire tampon cible et l'appel se termine avec CCWARN et le code anomalie RC2120. Notez que moins de TGTLEN octets peuvent être renvoyés dans ce cas.

- TGTLEN peut être inférieur ou supérieur à SRCLen. Au retour de MQXCNCV, DATLEN est inférieur ou égal à TGTLEN.

**Options de codage:** Les options suivantes peuvent être utilisées pour spécifier les codages de type entier des chaînes source et cible. Le codage approprié est utilisé uniquement lorsque l'identificateur de jeu de caractères correspondant indique que la représentation du jeu de caractères dans la mémoire principale dépend du codage utilisé pour les entiers binaires. Cela n'affecte que certains jeux de caractères multi-octets (par exemple, UTF-16 ).

Le codage est ignoré si le jeu de caractères est un jeu de caractères mono-octet (SBCS) ou un jeu de caractères multi-octets avec une représentation dans la mémoire principale qui ne dépend pas du codage d'entier.

Une seule des valeurs DCCS\* doit être spécifiée, combinée à l'une des valeurs DCCT\*:

**DCCSNA**

Le codage source est la valeur par défaut pour l'environnement et le langage de programmation.

**DCCSNO**

Le codage source est normal.

**DCCSRE**

Le codage source est inversé.

**DCCSUN**

Le codage source n'est pas défini.

**DCCTNA**

Le codage cible est la valeur par défaut pour l'environnement et le langage de programmation.

**DCCTNO**

Le codage cible est normal.

**DCCTRE**

Le codage cible est inversé.

## **DCCTUN**

Le codage cible n'est pas défini.

Les valeurs de codage définies précédemment peuvent être ajoutées directement à la zone OPTS . Toutefois, si le codage source ou cible est obtenu à partir de la zone MDENC dans le MQMD ou d'une autre structure, le traitement suivant doit être effectué:

1. Le codage d'entier doit être extrait de la zone MDENC en éliminant les codages à virgule flottante et décimal condensé ; pour plus de détails sur cette opération, voir «Analyse des codages sur IBM i», à la page 1504 .
2. Le codage d'entier résultant de l'étape 1 doit être multiplié par le facteur approprié avant d'être ajouté à la zone OPTS . Ces facteurs sont les suivants:

### **DCCSFA**

Facteur de codage de la source

### **DCCTFA**

Facteur pour le codage cible

S'il n'est pas spécifié, les options de codage sont définies par défaut sur undefined (DCC\* UN). Dans la plupart des cas, cela n'affecte pas la réussite de l'appel MQXCNV. Toutefois, si le jeu de caractères correspondant est un jeu de caractères multi-octets dont la représentation dépend du codage (par exemple, un jeu de caractères UTF-16 ), l'appel échoue avec le code anomalie RC2112 ou RC2116 , selon le cas.

**Option par défaut:** si aucune des options décrites précédemment n'est spécifiée, l'option suivante peut être utilisée:

### **DCCNON**

Aucune option spécifiée.

DCCNON est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette option soit utilisée avec une autre option, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

## **SRCCSI (entier signé à 10 chiffres)-entrée**

Identificateur de jeu de caractères codés de la chaîne avant la conversion.

Il s'agit de l'identificateur de jeu de caractères codés de la chaîne d'entrée dans SRCBUF.

## **SRCLLEN (entier signé à 10 chiffres)-entrée**

Longueur de la chaîne avant la conversion.

Il s'agit de la longueur en octets de la chaîne d'entrée dans SRCBUF ; il doit être supérieur ou égal à zéro.

## **SRCBUF (chaîne de caractères de 1 octet x SRCLLEN)-entrée**

Chaîne à convertir.

Il s'agit de la mémoire tampon contenant la chaîne à convertir d'un jeu de caractères à un autre.

## **TGTCSI (entier signé à 10 chiffres)-entrée**

Identificateur de jeu de caractères codés de la chaîne après la conversion.

Il s'agit de l'identificateur de jeu de caractères codés du jeu de caractères dans lequel SRCBUF doit être converti.

## **TGTLEN (entier signé à 10 chiffres)-entrée**

Longueur de la mémoire tampon de sortie.

Il s'agit de la longueur en octets de la mémoire tampon de sortie TGTBUF ; il doit être supérieur ou égal à zéro. Il peut être inférieur ou supérieur à SRCLLEN.

### **TGTBUF (chaîne de caractères de 1 octet x TGTLEN)-sortie**

Chaîne après conversion.

Il s'agit de la chaîne après sa conversion au jeu de caractères défini par TGTCSI. La chaîne convertie peut être plus courte ou plus longue que la chaîne non convertie. Le paramètre **DATLEN** indique le nombre d'octets valides renvoyés.

### **DATLEN (entier signé à 10 chiffres)-sortie**

Longueur de la chaîne de sortie.

Il s'agit de la longueur de la chaîne renvoyée dans la mémoire tampon de sortie TGTBUF. La chaîne convertie peut être plus courte ou plus longue que la chaîne non convertie.

### **CMPCOD (entier signé à 10 chiffres)-sortie**

Code achèvement.

Il peut s'agir de :

#### **CCOK**

Achèvement réussi.

#### **CCWARN**

Avertissement (achèvement partiel).

#### **CCFAIL**

Echec de l'appel.

### **REASON (entier signé à 10 chiffres)-sortie**

Code anomalie qualifiant CMPCOD.

Si CMPCOD est CCOK:

#### **RCNONE**

(0, X'000') Aucun code anomalie à signaler.

Si CMPCOD est CCWARN:

#### **RC2120**

(2120, X'848') Données converties trop volumineuses pour la mémoire tampon.

Si CMPCOD est CCFAIL:

#### **RC2010**

(2010, X'7DA') Paramètre de longueur des données non valide.

#### **RC2150**

(2150, X'866') Chaîne DBCS non valide.

#### **RC2018**

(2018, X'7E2') Descripteur de connexion non valide.

#### **RC2046**

(2046, X'7FE') Options incorrectes ou incohérentes.

#### **RC2102**

(2102, X'836') Ressources système disponibles insuffisantes.

#### **RC2145**

(2145, X'861') Paramètre de tampon source non valide.

#### **RC2111**

(2111, X'83F') Identificateur de jeu de caractères codés source non valide.

#### **RC2112**

(2112, X'840') Codage d'entier source non reconnu.

#### **RC2143**

(2143, X'85F') Paramètre de longueur source non valide.

**RC2071**

(2071, X'817') Mémoire disponible insuffisante.

**RC2146**

(2146, X'862') Paramètre de tampon cible non valide.

**RC2115**

(2115, X'843') Identificateur de jeu de caractères codés cible non valide.

**RC2116**

(2116, X'844') Codage d'entier cible non reconnu.

**RC2144**

(2144, X'860') Paramètre de longueur cible incorrect.

**RC2195**

(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes anomalie, voir [«Codes retour pour IBM i \(ILE RPG\)»](#), à la page 1498.

**Appel RPG (ILE)**

```

C*.1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQXCNCV(HCONN : OPTS : SRCCSI :
C                               SRCLEN : SRCBUF : TGTCSI :
C                               TGTLEN : TGTBUF : DATLEN :
C                               CMPCOD : REASON)

```

La définition de prototype pour l'appel est la suivante:

```

D*.1.....2.....3.....4.....5.....6.....7..
DMQXCNCV      PR          EXTPROC('MQXCNCV')
D* Connection handle
D HCONN              10I 0 VALUE
D* Options that control the action of MQXCNCV
D OPTS              10I 0 VALUE
D* Coded character set identifier of string before conversion
D SRCCSI            10I 0 VALUE
D* Length of string before conversion
D SRCLEN            10I 0 VALUE
D* String to be converted
D SRCBUF              *   VALUE
D* Coded character set identifier of string after conversion
D TGTCSI            10I 0 VALUE
D* Length of output buffer
D TGTLEN            10I 0 VALUE
D* String after conversion
D TGTBUF              *   VALUE
D* Length of output string
D DATLEN            10I 0
D* Completion code
D CMPCOD            10I 0
D* Reason code qualifying CMPCOD
D REASON            10I 0

```

**IBM i****MQCONVX (exit de conversion de données) sous IBM i**

Cette définition d'appel décrit les paramètres transmis à l'exit de conversion de données.

Aucun point d'entrée appelé MQCONVX n'est fourni par le gestionnaire de files d'attente (voir la note d'utilisation [«11»](#), à la page 1527).

Cette définition fait partie de l'interface de conversion de données IBM MQ (DCI), qui est l'une des interfaces de l'infrastructure IBM MQ.

- [«Syntaxe»](#), à la page 1526
- [«Remarques sur l'utilisation»](#), à la page 1526

- «Paramètres», à la page 1527
- «Appel RPG (ILE)», à la page 1529

## Syntaxe

(MQDXP, MQMD, INLEN, INBUF, OUTLEN, OUTBUF) MQCONVX

## Remarques sur l'utilisation

1. Un exit de conversion de données est un exit écrit par l'utilisateur qui reçoit le contrôle lors du traitement d'un appel MQGET. La fonction exécutée par l'exit de conversion de données est définie par le fournisseur de l'exit ; cependant, l'exit doit se conformer aux règles décrites ici et dans la structure de paramètres associée MQDXP.

Les langages de programmation pouvant être utilisés pour un exit de conversion de données sont déterminés par l'environnement.

2. L'exit est appelé uniquement si *toutes* les instructions suivantes sont vraies:
  - L'option GMCONV est spécifiée dans l'appel MQGET
  - La zone *MDFMT* du descripteur de message n'est pas FMNONE
  - Le message n'est pas déjà dans la représentation requise, c'est-à-dire que l'un ou les deux éléments *MDCSI* et *MDENC* du message sont différents de la valeur spécifiée par l'application dans le descripteur de message fourni dans l'appel MQGET.
  - Le gestionnaire de files d'attente n'a pas déjà effectué la conversion avec succès
  - La longueur de la mémoire tampon de l'application est supérieure à zéro
  - La longueur des données de message est supérieure à zéro
  - Le code anomalie jusqu'à présent lors de l'opération MQGET est RCNONE ou RC2079
3. Lors de l'écriture d'un exit, vous devez envisager de coder l'exit de manière à ce qu'il puisse convertir les messages qui ont été tronqués. Les messages tronqués peuvent se produire de l'une des manières suivantes:
  - L'application de réception fournit une mémoire tampon plus petite que le message, mais spécifie l'option GMATM sur l'appel MQGET.

Dans ce cas, la zone *DXREA* du paramètre **MQDXP** en entrée de l'exit aura la valeur RC2079.

- L'expéditeur du message l'a tronqué avant de l'envoyer. Cela peut se produire avec des messages de rapport, par exemple (voir «Conversion des messages de rapport sur IBM i», à la page 1514 pour plus de détails).

Dans ce cas, la zone *DXREA* du paramètre **MQDXP** en entrée de l'exit aura la valeur RCNONE (si l'application de réception a fourni une mémoire tampon suffisamment grande pour le message).

Par conséquent, la valeur de la zone *DXREA* en entrée de l'exit ne peut pas toujours être utilisée pour déterminer si le message a été tronqué.

La caractéristique distinctive d'un message tronqué est que la longueur fournie à l'exit dans le paramètre **INLEN** sera *inférieure* à la longueur impliquée par le nom de format contenu dans la zone *MDFMT* du descripteur de message. L'exit doit donc vérifier la valeur de **INLEN** avant de tenter de convertir les données ; l'exit *ne doit pas* supposer que la quantité totale de données impliquée par le nom de format a été fournie.

Si l'exit n'a pas été écrit pour convertir les messages tronqués et que **INLEN** est inférieur à la valeur attendue, l'exit doit renvoyer XRFAIL dans la zone *DXRES* du paramètre **MQDXP** , avec la zone *DXCC* définie sur CCWARN et la zone *DXREA* définie sur RC2110.

Si l'exit a été écrit pour convertir les messages tronqués, l'exit doit convertir autant de données que possible (voir la note d'utilisation suivante), en prenant soin de ne pas tenter d'examiner ou de convertir les données au-delà de la fin de *INBUF*. Si la conversion aboutit, l'exit doit laisser la zone *DXREA* dans le paramètre **MQDXP** inchangée. Cette commande renvoie RC2079 si le message a été

tronqué par le gestionnaire de files d'attente du récepteur et RCNONE si le message a été tronqué par l'expéditeur du message.

Il est également possible pour un message de développer *lors de la conversion*, jusqu'à ce qu'il soit supérieur à *OUTBUF*. Dans ce cas, l'exit doit décider si le message doit être tronqué ; la zone *DXAOP* dans le paramètre **MQDXP** indique si l'application de réception a spécifié l'option GMATM.

4. En règle générale, il est recommandé de convertir toutes les données du message fourni à l'exit dans *INBUF*, ou qu'aucune de ces données ne soit convertie. Une exception à cette règle se produit cependant si le message est tronqué, soit avant la conversion, soit pendant la conversion ; dans ce cas, il peut y avoir un élément incomplet à la fin de la mémoire tampon (par exemple: un octet d'un caractère codé sur deux octets, ou 3 octets d'un entier codé sur quatre octets). Dans cette situation, il est recommandé d'omettre l'élément incomplet et de définir les valeurs nulles pour les octets inutilisés dans *OUTBUF*. Toutefois, les éléments ou caractères complets d'un tableau ou d'une chaîne *doivent* être convertis.
5. Lorsqu'un exit est nécessaire pour la première fois, le gestionnaire de files d'attente tente de charger un objet portant le même nom que le format (à l'exception des extensions). L'objet chargé doit contenir l'exit qui traite les messages avec ce nom de format. Il est recommandé que le nom de l'exit et le nom de l'objet qui contient l'exit soient identiques, bien que tous les environnements ne l'exigent pas.
6. Une nouvelle copie de l'exit est chargée lorsqu'une application tente d'extraire le premier message qui utilise cette *MDFMT* depuis que l'application s'est connectée au gestionnaire de files d'attente. Une nouvelle copie peut également être chargée à d'autres moments, si le gestionnaire de files d'attente a supprimé une copie précédemment chargée. Pour cette raison, un exit ne doit pas tenter d'utiliser la mémoire statique pour communiquer des informations d'un appel de l'exit au suivant-l'exit peut être déchargé entre les deux appels.
7. S'il existe un exit fourni par l'utilisateur portant le même nom que l'un des formats intégrés pris en charge par le gestionnaire de files d'attente, l'exit fourni par l'utilisateur ne remplace pas la routine de conversion intégrée. Les seules circonstances dans lesquelles un tel exit est appelé sont les suivantes:
  - Si la routine de conversion intégrée ne peut pas gérer les conversions vers ou depuis le *MDCSI* ou *MDENC* impliqué, ou
  - Si la routine de conversion intégrée n'a pas réussi à convertir les données (par exemple, parce qu'une zone ou un caractère ne peut pas être converti).
8. La portée de l'exit dépend de l'environnement. Les noms *MDFMT* doivent être choisis de manière à réduire le risque de conflits avec d'autres formats. Il est recommandé de commencer par les caractères qui identifient l'application définissant le nom de format.
9. L'exit de conversion de données s'exécute dans un environnement tel que celui du programme qui a émis l'appel MQGET ; l'environnement inclut l'espace adresse et le profil utilisateur (le cas échéant). Le programme peut être un agent de canal de transmission de messages envoyant des messages à un gestionnaire de files d'attente de destination qui ne prend pas en charge la conversion de messages. L'exit ne peut pas compromettre l'intégrité du gestionnaire de files d'attente, car il ne s'exécute pas dans l'environnement du gestionnaire de files d'attente.
10. Le seul appel MQI pouvant être utilisé par l'exit est MQXCNCV ; la tentative d'utilisation d'autres appels MQI échoue avec le code anomalie RC2219 ou d'autres erreurs imprévisibles.
11. Aucun point d'entrée appelé MQCONVX n'est fourni par le gestionnaire de files d'attente. Le nom de l'exit doit être identique au nom de format (nom contenu dans la zone *MDFMT* de MQMD), bien que cela ne soit pas obligatoire dans tous les environnements.

## Paramètres

L'appel MQCONVX comporte les paramètres suivants:

### **MQDXP (MQDXP)-entrée/sortie**

Bloc de paramètres d'exit de conversion de données.

Cette structure contient des informations relatives à l'appel de l'exit. L'exit définit des informations dans cette structure pour indiquer le résultat de la conversion. Pour plus de détails sur les zones de cette structure, voir «MQDXP (paramètre d'exit de conversion de données) sous IBM i», à la page 1515.

### **MQMD (MQMD)-entrée/sortie**

Descripteur de message.

Lors de l'entrée de l'exit, il s'agit du descripteur de message qui serait renvoyé à l'application si aucune conversion n'était effectuée. Il contient donc les *MDFMT*, *MDENC* et *MDCSI* du message non converti contenu dans *INBUF*.

**Remarque :** Le paramètre **MQMD** transmis à l'exit est toujours la version la plus récente de MQMD prise en charge par le gestionnaire de files d'attente qui appelle l'exit. Si l'exit est destiné à être portable entre différents environnements, il doit vérifier la zone *MDVER* dans *MQMD* pour vérifier que les zones auxquelles l'exit doit accéder sont présentes dans la structure.

Sous IBM i, l'exit est transmis à un MQMD version-2.

En sortie, l'exit doit remplacer les zones *MDENC* et *MDCSI* par les valeurs demandées par l'application, si la conversion a abouti ; ces modifications seront répercutées sur l'application. Toutes les autres modifications apportées par l'exit à la structure sont ignorées ; elles ne sont pas reflétées dans l'application.

Si l'exit renvoie XROK dans la zone *DXRES* de la structure MQDXP, mais ne modifie pas les zones *MDENC* ou *MDCSI* du descripteur de message, le gestionnaire de files d'attente renvoie pour ces zones les valeurs que les zones correspondantes de la structure MQDXP avaient en entrée de l'exit.

### **INLEN (entier signé à 10 chiffres)-entrée**

Longueur en octets de *INBUF*.

Il s'agit de la longueur de la mémoire tampon d'entrée *INBUF*, qui indique le nombre d'octets à traiter par l'exit. *INLEN* est la longueur la moins élevée des données de message avant la conversion et la longueur de la mémoire tampon fournie par l'application lors de l'appel MQGET.

La valeur est toujours supérieure à zéro.

### **INBUF (chaîne de 1 octet x INLEN)-entrée**

Mémoire tampon contenant le message non converti.

Contient les données du message avant la conversion. Si l'exit ne parvient pas à convertir les données, le gestionnaire de files d'attente renvoie le contenu de cette mémoire tampon à l'application une fois l'exit terminé.

**Remarque :** L'exit ne doit pas modifier *INBUF* ; si ce paramètre est modifié, les résultats ne sont pas définis.

### **OUTLEN (entier signé à 10 chiffres)-entrée**

Longueur en octets de *OUTBUF*.

Il s'agit de la longueur de la mémoire tampon de sortie *OUTBUF*, qui est identique à la longueur de la mémoire tampon fournie par l'application sur l'appel MQGET.

La valeur est toujours supérieure à zéro.

### **OUTBUF (chaîne de 1 octet x OUTLEN)-sortie**

Mémoire tampon contenant le message converti.

En sortie de l'exit, si la conversion a abouti (comme indiqué par la valeur XROK dans la zone *DXRES* du paramètre **MQDXP**), **OUTBUF** contient les données de message à distribuer à l'application, dans la représentation demandée. Si la conversion a échoué, les modifications apportées par l'exit à cette mémoire tampon sont ignorées.



## Appel RPG (ILE)

```
C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      exitname(MQDXP : MQMD : INLEN :
C                               INBUF : OUTLEN : OUTBUF)
```

La définition de prototype pour l'appel est la suivante:

```
D*..1.....2.....3.....4.....5.....6.....7..
Dexitname      PR          EXTPROC('exitname')
D* Data-conversion exit parameter block
D MQDXP                44A
D* Message descriptor
D MQMD                364A
D* Length in bytes of INBUF
D INLEN                10I 0 VALUE
D* Buffer containing the unconverted message
D INBUF                *   VALUE
D* Length in bytes of OUTBUF
D OUTLEN              10I 0 VALUE
D* Buffer containing the converted message
D OUTBUF                *   VALUE
```

**Fin de l'interface de programmation sensible au produit**

## Références des exits utilisateur, des exits API et des services installables

Utilisez les informations de la présente section pour vous aider à développer vos exits utilisateur, vos exits API et vos applications de services installables:

- [«Structure MQIEP», à la page 1529](#)
- [«Référence de l'exit de conversion de données», à la page 1533](#)
- [«MQ\\_PUBLISH\\_EXIT-Exit de publication», à la page 1537](#)
- [«Structures de données et appels d'exit de canal», à la page 1545](#)
- [«Référence d'exit API», à la page 1638](#)
- [«Informations de référence de l'interface des services installables», à la page 1699](#)

### Concepts associés

[Exits utilisateur, exits API et services optionnels d'IBM MQ](#)

### Tâches associées

[Extension des fonctions du gestionnaire de files d'attente](#)

## Structure MQIEP

La structure MQIEP contient un point d'entrée pour chaque appel de fonction que les exits sont autorisés à effectuer.

### Zones

#### StrucId

Type: MQCHAR4 -entrée

Identificateur de structure. La valeur est la suivante:

**ID\_STRUC\_MQIPE**

#### Version

Type : MQLONG - entrée

Numéro de version de la structure. La valeur est la suivante:

**MQIEP\_VERSION\_1**

Numéro de version de la structure de la version 1.

**MQIEP\_VERSION\_CURRENT\_**

Version actuelle de la structure.

**StrucLength**

Type: MQLONG

Taille de la structure MQIEP en octets. La valeur est la suivante:

**MQIEP\_LENGTH\_1****Indicateurs**

Type: MQLONG

Fournit des informations sur les adresses de fonction. Un indicateur pour indiquer si la bibliothèque est à unités d'exécution peut être utilisé avec un indicateur pour indiquer si la bibliothèque est une bibliothèque client ou serveur.

La valeur suivante est utilisée pour ne spécifier aucune information de bibliothèque:

**MQIEPF\_AUCUN**

L'une des valeurs suivantes est utilisée pour indiquer si la bibliothèque partagée est à unités d'exécution ou non:

**MQIEPF\_BIBLIOTHÈQUE\_NON\_UNITÉ\_UNITÉ\_EXÉCUTION**

Une bibliothèque partagée sans unités d'exécution

**MQIEPF\_BIBLIOTHEQUE à unités d'exécution**

Une bibliothèque partagée à unités d'exécution

L'une des valeurs suivantes est utilisée pour indiquer si la bibliothèque partagée est un client ou une bibliothèque partagée de serveur:

**BIBLIOTHÈQUE\_CLIENT\_MQIEPF**

Une bibliothèque partagée client

**MQIEPF\_BIBLIOTHÈQUE\_LOCALE**

Une bibliothèque partagée de serveur

**Réservé**

Type: MQPTR

**Appel MQBACK\_Call**

Type: PMQ\_BACK\_CALL

Adresse de l'appel MQBACK.

**Appel MQBEGIN\_Call**

Type: PMQ\_BEGIN\_CALL

Adresse de l'appel MQBEGIN.

**Appel MQBUFMH\_Call**

Type: PMQ\_BUFMH\_CALL

Adresse de l'appel MQBUFMH.

**Appel\_MQCB**

Type: PMQ\_CB\_CALL

Adresse de l'appel MQCB.

**Appel MQCLOSE\_Call**

Type: PMQ\_CLOSE\_CALL

Adresse de l'appel MQCLOSE.

**Appel\_MQCM**

Type: PMQ\_CMIT\_CALL

Adresse de l'appel MQCMIT.

**Appel MQCONN\_Call**

Type: PMQ\_CONN\_CALL

Adresse de l'appel MQCONN.

**Appel MQCONNX\_Call**

Type: PMQ\_CONNX\_CALL

Adresse de l'appel MQCONNX.

**MQCRTMH\_Appel**

Type: PMQ\_CRTMH\_CALL

Adresse de l'appel MQCRTMH.

**Appel MQCTL**

Type: PMQ\_CTL\_CALL

Adresse de l'appel MQCTL.

**Appel MQDC**

Type: PMQ\_DISC\_CALL

Adresse de l'appel MQDISC.

**Appel MQDLTMH\_Call**

Type: PMQ\_DLTMH\_CALL

Adresse de l'appel MQDLTMH.

**Appel MQDLTMP\_Call**

Type: PMQ\_DLTMP\_CALL

Adresse de l'appel MQDLTMP.

**Appel MQGET**

Type: PMQ\_GET\_CALL

Adresse de l'appel MQGET.

**Appel MQINQ\_Call**

Type: PMQ\_INQ\_CALL

Adresse de l'appel MQINQ.

**Appel MQINQMP\_Call**

Type: PMQ\_INQMP\_CALL

Adresse de l'appel MQINQMP.

**Appel MQMHBUF\_Appel**

Type: PMQ\_MHBUF\_CALL

Adresse de l'appel MQMHBUF.

**Appel MQOPEN\_Call**

Type: PMQ\_OPEN\_CALL

Adresse de l'appel MQOPEN.

**Appel MQPUT\_Call**

Type: PMQ\_PUT\_CALL

Adresse de l'appel MQPUT.

**MQPUT1\_Call**

Type: PMQ\_PUT1\_CALL

Adresse de l'appel MQPUT1 .

**Appel MQSET\_Call**

Type: PMQ\_SET\_CALL

Adresse de l'appel MQSET.

**Appel MQSETMP\_Call**

Type: PMQ\_SETMP\_CALL

Adresse de l'appel MQSETMP.

**Appel\_MQSTAT\_Appel**

Type: PMQ\_STAT\_CALL

Adresse de l'appel MQSTAT.

**MQSUB\_Appel**

Type: PMQ\_SUB\_CALL

Adresse de l'appel MQSUB.

**Appel MQSUBRQ\_Call**

Type: PMQ\_SUBRQ\_CALL

Adresse de l'appel MQSUBRQ.

**Appel MQXCNVC\_Call**

Type: PMQ\_XCNVC\_CALL

Adresse de l'appel MQXCNVC.

**Appel MQXCLWLN\_Call**

Type: PMQ\_XCLWLN\_CALL

Adresse de l'appel MQXCLWLN.

**Appel MQXDX\_Call**

Type: PMQ\_XDX\_CALL

Adresse de l'appel MQXDX.

**Appel MQXEP\_Call**

Type: PMQ\_XEP\_CALL

Adresse de l'appel MQXEP.

**Appel\_MQZ**

Type: PMQ\_ZEP\_CALL

Adresse de l'appel MQZEP.

**Déclaration C**

```

struct tagMQIEP {
    MQCHAR4      StrucId;           /* Structure identifier */
    MQLONG       Version;          /* Structure version number */
    MQLONG       StructLength;     /* Structure length */
    MQLONG       Flags;           /* Flags */
    MQPTR        Reserved;        /* Reserved */
    PMQ_BACK_CALL MQBACK_Call;    /* Address of MQBACK */
    PMQ_BEGIN_CALL MQBEGIN_Call;  /* Address of MQBEGIN */
    PMQ_BUFMH_CALL MQBUFMH_Call;  /* Address of MQBUFMH */
    PMQ_CB_CALL  MQCB_Call;       /* Address of MQCB */
    PMQ_CLOSE_CALL MQCLOSE_Call;  /* Address of MQCLOSE */
    PMQ_COMMIT_CALL MQCMIT_Call;  /* Address of MQCMIT */
    PMQ_CONN_CALL MQCONN_Call;    /* Address of MQCONN */
    PMQ_CONNX_CALL MQCONNX_Call;  /* Address of MQCONNX */
    PMQ_CRTMH_CALL MQCRTMH_Call;  /* Address of MQCRTMH */
    PMQ_CTL_CALL  MQCTL_Call;     /* Address of MQCTL */
    PMQ_DISC_CALL MQDISC_Call;    /* Address of MQDISC */
    PMQ_DLTMH_CALL MQDLTMH_Call;  /* Address of MQDLTMH */
    PMQ_DLTMP_CALL MQDLTMP_Call;  /* Address of MQDLTMP */
    PMQ_GET_CALL  MQGET_Call;     /* Address of MQGET */
}

```

```

PMQ_INQ_CALL      MQINQ_Call;      /* Address of MQINQ */
PMQ_INQMP_CALL   MQINQMP_Call; /* Address of MQINQMP */
PMQ_MHBUF_CALL   MQMHBUF_Call; /* Address of MQMHBUF */
PMQ_OPEN_CALL    MQOPEN_Call; /* Address of MQOPEN */
PMQ_PUT_CALL     MQPUT_Call; /* Address of MQPUT */
PMQ_PUT1_CALL    MQPUT1_Call; /* Address of MQPUT1 */
PMQ_SET_CALL     MQSET_Call; /* Address of MQSET */
PMQ_SETMP_CALL   MQSETMP_Call; /* Address of MQSETMP */
PMQ_STAT_CALL    MQSTAT_Call; /* Address of MQSTAT */
PMQ_SUB_CALL     MQSUB_Call; /* Address of MQSUB */
PMQ_SUBRQ_CALL   MQSUBRQ_Call; /* Address of MQSUBRQ */
PMQ_XCLWLN_CALL  MQXCLWLN_Call; /* Address of MQXCLWLN */
PMQ_XCNVC_CALL   MQXCNVC_Call; /* Address of MQXCNVC */
PMQ_XDX_CALL     MQXDX_Call; /* Address of MQXDX */
PMQ_XEP_CALL     MQXEP_Call; /* Address of MQXEP */
PMQ_ZEP_CALL     MQZEP_Call; /* Address of MQZEP */
};

```



## Référence de l'exit de conversion de données

Pour z/OS, vous devez écrire des exits de conversion de données en langage assembleur. Pour les autres plateformes, il est recommandé d'utiliser le langage de programmation C.

Pour vous aider à créer un programme d'exit de conversion de données, les ressources suivantes sont fournies:

- Un fichier source squelette
- Un appel de conversion de caractères
- Utilitaire qui crée un fragment de code qui effectue la conversion de données sur les structures de type de données. Cet utilitaire utilise uniquement l'entrée C. Sous z/OS, il génère du code assembleur.

Pour la procédure d'écriture des programmes, voir:

-  [Ecriture d'un programme d'exit de conversion de données pour IBM MQ for IBM i](#)
-  [Ecriture d'un programme d'exit de conversion de données pour IBM MQ for z/OS](#)
- [Ecriture d'un exit de conversion de données pour les systèmes IBM MQ for AIX or Linux](#)
- [Ecriture d'un exit de conversion de données pour IBM MQ for Windows](#)

### Fichier source squelette

Ils peuvent être utilisés comme point de départ lors de l'écriture d'un programme d'exit de conversion de données.

Les fichiers fournis sont répertoriés dans [Tableau 816](#), à la page 1533.






<i>Tableau 816. Fichiers source de squelette</i>	
Plateforme	Fichier
 AIX	amqsvfc0.c
 IBM i	QMQMSAMP/QCSRC (AMQSVFC4)
 Linux	amqsvfc0.c
Systemes  Windows	amqsvfc0.c
 z/OS	CSQ4BAX8 ( «1», à la page 1534 ) CSQ4BAX9 ( «2», à la page 1534 ) CSQ4CAX9 ( «3», à la page 1534 )

Tableau 816. Fichiers source de squelette (suite)

Plateforme	Fichier
<b>Remarques :</b>	
<ol style="list-style-type: none"> <li>1. Illustre l'appel MQXCVNC.</li> <li>2. Un encapsuleur pour les fragments de code générés par l'utilitaire à utiliser dans tous les environnements, à l'exception de CICS.</li> <li>3. Encapsuleur pour les fragments de code générés par l'utilitaire pour une utilisation dans l'environnement CICS .</li> </ol>	

## Appel de conversion de caractères

Utilisez l'appel MQXCNVC (conversion de caractères) à partir d'un programme d'exit de conversion de données pour convertir des données de message de type caractères d'un jeu de caractères à un autre. Pour certains jeux de caractères multi-octets (par exemple, les jeux de caractères UTF-16 ), les options appropriées doivent être utilisées.

Aucun autre appel MQI ne peut être effectué à partir de l'exit ; la tentative d'exécution d'un tel appel échoue avec le code anomalie MQRC\_CALL\_IN\_PROGRESS.

Voir «MQXCNVC-Conversion de caractères», à la page 964 pour plus d'informations sur l'appel MQXCNVC et les options appropriées.

## Utilitaire permettant de créer un code de sortie de conversion

Utilisez ces informations pour en savoir plus sur la création d'un code de sortie de conversion.

Les commandes permettant de créer un code d'exit de conversion sont les suivantes:

**IBM i**  
CVTMQMDTA (Convertir le type de données IBM MQ )

**ALW**  
crtmqcvx (Créer IBM MQ conversion-exit)

**z/OS**  
CSQUCVX

La commande de votre plateforme génère un fragment de code qui effectue la conversion de données sur les structures de type de données, à utiliser dans votre programme d'exit de conversion de données. La commande utilise un fichier contenant une ou plusieurs définitions de structure de langage C.

**z/OS** Sous z/OS, il génère ensuite un fichier contenant des fragments de code assembleur et des fonctions de conversion. Sur les autres plateformes, il génère un fichier avec une fonction C pour convertir chaque définition de structure. Sous z/OS, l'utilitaire requiert l'accès à la bibliothèque d'exécution LE/370 SCEERUN.

## Appel de l'utilitaire CSQUCVX sous z/OS

**z/OS**

La [Figure 10](#), à la page 1535 illustre un exemple de JCL utilisé pour appeler l'utilitaire CSQUCVX.

```
//CVX EXEC PGM=CSQUCVX
//STEPLIB DD DISP=SHR,DSN=thlqua1.SCSQANLE
// DD DISP=SHR,DSN=thlqua1.SCSQLOAD
// DD DISP=SHR,DSN=le370qua1.SCEERUN
//SYSPRINT DD SYSOUT=*
//CSQUINP DD DISP=SHR,DSN=MY.MQSERIES.FORMATS(MSG1)
//CSQUOUT DD DISP=OLD,DSN=MY.MQSERIES.EXIT(S(MSG1)
```

Figure 10. Exemple de JCL utilisé pour appeler l'utilitaire CSQUCVX

## Instructions de définition de données z/OS



L'utilitaire CSQUCVX requiert des instructions de définition de données avec les noms de définition de données suivants affichés dans [Tableau 817](#), à la page 1535:

Tableau 817. Noms et descriptions des instructions de définition de données	
instruction de définition de données	Description
SYSPRINT	Indique un fichier ou une classe de spoule d'impression pour les rapports et les messages d'erreur.
CSQUINP	Indique le fichier partitionné contenant les définitions des structures de données à convertir.
CSQUOUT	Indique le fichier partitionné dans lequel les fragments de code de conversion doivent être écrits. La longueur d'enregistrement logique (LRECL) doit être 80 et le format d'enregistrement (RECFM) doit être FB.

## Messages d'erreur dans les systèmes AIX, Linux, and Windows

La commande `crtmqcvx` renvoie des messages compris entre AMQ7953 et AMQ7970.

Ces messages sont répertoriés dans [Messages et codes anomalie IBM MQ Messages](#).

Il existe deux principaux types d'erreur:

- Les erreurs majeures, telles que les erreurs de syntaxe, lorsque le traitement ne peut pas continuer.  
Un message s'affiche à l'écran indiquant le numéro de ligne de l'erreur dans le fichier d'entrée. Le fichier de sortie a peut-être été partiellement créé.
- Autres erreurs lorsqu'un message s'affiche indiquant qu'un problème a été détecté mais que l'analyse syntaxique de la structure peut continuer.

Le fichier de sortie a été créé et contient des informations d'erreur sur les problèmes qui se sont produits. Ces informations d'erreur sont préfixées par `#error` de sorte que le code généré ne soit accepté par aucun compilateur sans intervention pour corriger les problèmes.

## Syntaxe valide

Votre fichier d'entrée pour l'utilitaire doit être conforme à la syntaxe du langage C.

Si vous n'êtes pas familiarisé avec C, reportez-vous à l' [exemple C](#) dans cette rubrique.

En outre, tenez compte des règles suivantes:

- `typedef` est reconnu uniquement avant le mot clé `struct`.
- Une balise de structure est requise sur vos déclarations de structure.

- Vous pouvez utiliser des crochets vides [ ] pour indiquer un tableau ou une chaîne de longueur variable à la fin d'un message.
- Les tableaux multidimensionnels et les tableaux de chaînes ne sont pas pris en charge.
- Les types de données supplémentaires suivants sont reconnus:
  - MQBOOL
  - MQBYTE
  - MQCHAR
  - MQFLOAT32
  - MQFLOAT64
  - MQSHORT
  - MQLONG
  - MQINT8
  - MQUINT8
  - MQINT16
  - MQUINT16
  - MQINT32
  - MQUINT32
  - MQINT64
  - MQUINT64

Les zones MQCHAR sont converties en page de codes, mais MQBYTE, MQINT8 et MQUINT8 restent intactes. Si le codage est différent, MQSHORT, MQLONG, MQINT16, MQUINT16, MQINT32, MQUINT32, MQINT64, MQUINT64, MQFLOAT32, MQFLOAT64 et MQBOOL sont convertis en conséquence.

- N'utilisez pas les types de données suivants:
  - double
  - Pointeurs
  - zones de bits

En effet, l'utilitaire de création de code de sortie de conversion ne permet pas de convertir ces types de données. Pour y remédier, vous pouvez écrire vos propres routines et les appeler à partir de l'exit.

Autres points à noter:

- N'utilisez pas de numéros de séquence dans le fichier d'entrée.
- S'il existe des zones pour lesquelles vous souhaitez fournir vos propres routines de conversion, déclarez-les comme MQBYTE, puis remplacez les macros CMQXCFBA générées par votre propre code de conversion.

## Exemple C

```
struct TEST { MQLONG    SERIAL_NUMBER;
             MQCHAR    ID[5];
             MQINT16   VERSION;
             MQBYTE    CODE[4];
             MQLONG    DIMENSIONS[3];
             MQCHAR    NAME[24];
             } ;
```

Cela correspond aux déclarations suivantes dans d'autres langages de programmation:



## COBOL

```
10 TEST.  
15 SERIAL-NUMBER PIC S9(9) BINARY.  
15 ID PIC X(5).  
15 VERSION PIC S9(4) BINARY.  
* CODE IS NOT TO BE CONVERTED  
15 CODE PIC X(4).  
15 DIMENSIONS PIC S9(9) BINARY OCCURS 3 TIMES.  
15 NAME PIC X(24).
```

## System/390

```
TEST EQU *  
SERIAL_NUMBER DS F  
ID DS CL5  
VERSION DS H  
CODE DS XL4  
DIMENSIONS DS 3F  
NAME DS CL24
```

## PL/I

### Pris en charge sur z/OS uniquement

```
DCL 1 TEST,  
2 SERIAL_NUMBER FIXED BIN(31),  
2 ID CHAR(5),  
2 VERSION FIXED BIN(15),  
2 CODE CHAR(4), /* not to be converted */  
2 DIMENSIONS(3) FIXED BIN(31),  
2 NAME CHAR(24);
```

## MQ\_PUBLISH\_EXIT-Exit de publication

L'appel MQ\_PUBLISH\_EXIT peut inspecter et modifier les messages distribués aux abonnés.

### Objet

Utilisez l'exit de publication pour inspecter et modifier les messages distribués aux abonnés:

- Examiner le contenu d'un message publié pour chaque abonné
- Modifier le contenu d'un message publié pour chaque abonné
- Modifier la file d'attente dans laquelle un message est inséré
- Arrêter la distribution d'un message à un abonné

Cet exit n'est pas disponible sous IBM MQ for z/OS.

### Syntaxe

**MQ\_PUBLISH\_EXIT** (*ExitParms*, *PubContext*, *SubContext*)

### Paramètres

#### **ExitParms (MQPSXP)** - Input/Output

*ExitParms* contient des informations sur l'appel de l'exit.

#### **PubContext (MQPBC)** - Input

*PubContext* contient des informations contextuelles sur le diffuseur de publications de la publication.

## SubContext (MQSBC) - Input/Output

*SubContext* contient des informations contextuelles sur l'abonné recevant la publication.

## MQPSXP-Structure de données de l'exit de publication

La structure MQPSXP décrit les informations qui sont transmises à et renvoyées par l'exit de publication.

Le Tableau 818, à la page 1538 récapitule les zones de la structure:

Tableau 818. Zones dans MQPSXP	
Zone	Description
<u>StrucID</u>	Identificateur de structure
<u>Version</u>	Numéro de version de structure
<u>ExitId</u>	Type d'exit appelé
<u>ExitReason</u>	Raison de l'appel de l'exit
<u>ExitResponse</u>	Réponse de l'exit
<u>ExitResponse2</u>	Réponse secondaire de l'exit
<u>Feedback</u>	Code retour
<u>ExitUserArea</u>	Quitter la zone utilisateur
<u>ExitData</u>	Données d'exit
<u>QMgrName</u>	Nom du gestionnaire de files d'attente local
<u>Hconn</u>	Descripteur de connexion
<u>MsgDescPtr</u>	Adresse du descripteur de message (MQMD)
<u>MsgHandle</u>	Traitement des propriétés de message (MQHMSG)
<u>MsgInPtr</u>	Adresse du message d'entrée
<u>MsgInLength</u>	Longueur du message d'entrée
<u>MsgOutPtr</u>	Adresse du message de sortie
<u>MsgOutLength</u>	Longueur du message de sortie
<u>pEntryPoints</u>	Adresse de la structure MQIEP

## Zones

### StrucID (MQCHAR4)

*StrucID* est l'identificateur de structure. La valeur est la suivante:

#### MQPSXP\_STRUCID

MQPSXP\_STRUCID est l'identificateur de la structure des paramètres d'exit de publication. Pour le langage de programmation C, la constante MQPSXP\_STRUC\_ID\_ARRAY est également définie ; elle a la même valeur que MQPSXP\_STRUC\_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

*StrucID* est une zone d'entrée de l'exit.

### Version (MQLONG)

*Version* est le numéro de version de la structure. La valeur est la suivante:

#### MQPSXP\_VERSION\_1

MQPSXP\_VERSION\_1 est la structure des paramètres d'exit de publication version 1. La constante MQPSXP\_CURRENT\_VERSION est également définie avec la même valeur.

*Version* est une zone d'entrée de l'exit.

### **ExitId (MQLONG)**

*ExitId* est le type d'exit appelé. La valeur est la suivante:

#### **MQXT\_PUBLISH\_EXIT**

Exit de publication.

*ExitId* est une zone d'entrée de l'exit.

### **ExitReason (MQLONG)**

*ExitReason* est la raison pour laquelle l'exit est appelé. Les valeurs possibles sont les suivantes:

#### **MQXR\_INIT**

L'exit de cette connexion est appelé pour l'initialisation. L'exit peut acquérir et initialiser les ressources dont il a besoin ; par exemple, la mémoire principale.

#### **MQXR\_TERM**

L'exit de cette connexion est appelé car l'exit est sur le point d'être arrêté. L'exit doit libérer toutes les ressources qu'il a acquises depuis qu'il a été initialisé ; par exemple, la mémoire principale.

#### **MQXR\_PUBLICATION**

L'exit est appelé par le gestionnaire de files d'attente avant de placer une publication dans une file d'attente de messages d'un abonné. L'exit peut modifier le message, ne pas le placer dans la file d'attente ou arrêter la publication.

*ExitReason* est une zone d'entrée de l'exit.

### **ExitResponse (MQLONG)**

Définissez *ExitResponse* dans l'exit pour indiquer comment le traitement doit se poursuivre.

*ExitResponse* est l'une des valeurs suivantes:

#### **MQXCC\_OK**

Définissez MQXCC\_OK pour poursuivre le traitement normalement. Définissez MQXCC\_OK en réponse à toute valeur de *ExitReason*.

Si *ExitReason* a la valeur MQXR\_PUBLICATION, les zones *DestinationQName* et *DestinationQMgrName* de la structure MQSBC identifient la destination à laquelle le message est envoyé.

#### **MQXCC\_FAILED**

Définissez MQXCC\_FAILED pour arrêter l'opération de publication. Le code achèvement MQCC\_FAILED et le code anomalie 2557 (09FD) (RC2557): MQRC\_PUBLISH\_EXIT\_ERROR sont définis en cas de retour de l'exit.

#### **MQXCC\_SUPPRESS\_FUNCTION**

Définissez MQXCC\_SUPPRESS\_FUNCTION pour arrêter le traitement normal du message. Définissez MQXCC\_SUPPRESS\_FUNCTION uniquement si *ExitReason* a la valeur MQXR\_PUBLICATION.

Le message continue d'être traité par le gestionnaire de files d'attente en fonction de l'option MQRO\_DISCARD\_MSG dans la zone *Report* du descripteur de message du message.

- Si l'option MQRO\_DISCARD\_MSG est spécifiée, le message n'est pas distribué à l'abonné.
- Si l'option MQRO\_DISCARD\_MSG n'est pas spécifiée, le message est placé dans la file d'attente de rebut. S'il n'y a pas de file d'attente de rebut ou si le message ne peut pas être placé dans la file d'attente de rebut, la publication n'est pas distribuée à l'abonné. La distribution de la publication aux autres abonnés dépend des valeurs des attributs d'objet de rubrique PMSGDLV et NPMSGDLV. Pour plus d'informations sur ces attributs, voir les descriptions des paramètres de la commande DEFINE TOPIC.

*ExitResponse* est une zone de sortie de l'exit.

### **ExitResponse2 (MQLONG)**

*ExitResponse2* est réservé pour une utilisation ultérieure.

### **Feedback (MQLONG)**

*Feedback* est le code retour à utiliser si l'exit renvoie MQXCC\_SUPPRESS\_FUNCTION dans *ExitResponse*.

En entrée de l'exit, *Feedback* a toujours la valeur MQFB\_NONE. Si l'exit renvoie MQXCC\_SUPPRESS\_FUNCTION, définissez *Feedback* sur la valeur à utiliser pour le message lorsque le gestionnaire de files d'attente le place dans la file d'attente de rebut. En retour de l'exit, si *Feedback* a la valeur d'origine MQFB\_NONE, le gestionnaire de files d'attente définit *Feedback* sur MQFB\_STOPPED\_BY\_PUBSUB\_EXIT.

*Feedback* est une zone d'entrée-sortie de l'exit.

### **ExitUserArea (MQBYTE16)**

*ExitUserArea* est une zone disponible que l'exit peut utiliser. Chaque connexion possède un *ExitUserArea* distinct. La longueur de *ExitUserArea* est donnée par MQ\_EXIT\_USER\_AREA\_LENGTH.

La zone *ExitReason* a la valeur MQXR\_INIT lors du premier appel de l'exit. *ExitUserArea* est initialisé sur MQXUA\_NONE lors du premier appel de l'exit pour une connexion. Les modifications ultérieures apportées à *ExitUserArea* sont conservées entre les appels de l'exit.

*ExitUserArea* est une zone d'entrée-sortie de l'exit.

### **ExitData (MQCHAR32)**

*ExitData* est une donnée d'exit fixe définie par le paramètre **PublishExitData** de la strophe dans le fichier d'initialisation du gestionnaire de files d'attente. Les données sont remplies avec des blancs sur toute la longueur de la zone. Si aucune donnée d'exit fixe n'est définie dans le fichier d'initialisation, *ExitData* est vide. La longueur de *ExitData* est donnée par MQ\_EXIT\_DATA\_LENGTH.

*ExitData* est une zone d'entrée de l'exit.

### **QMgrName (MQCHAR48)**

*QMgrName* est le nom du gestionnaire de files d'attente local. Le nom est rempli avec des blancs sur toute la longueur de la zone. La longueur de cette zone est indiquée par MQ\_Q\_MGR\_NAME\_LENGTH.

*QMgrName* est une zone d'entrée de l'exit.

### **Hconn (MQHCONN)**

*Hconn* est le descripteur représentant une connexion au gestionnaire de files d'attente. Utilisez uniquement *Hconn* comme paramètre pour les appels de fonction de propriété de message MQSETMP, MQINQMMP ou MQDLTMP pour utiliser les propriétés de message.

*Hconn* est une zone d'entrée de l'exit.

### **MsgDescPtr (PMQMD)**

*MsgDescPtr* est l'adresse du descripteur de message (MQMD) du message en cours de traitement et est une copie du MQMD renvoyé par l'appel MQPUT. L'exit peut modifier le contenu du descripteur de message. Toute modification du contenu du descripteur de message doit être effectuée avec précaution. En particulier, dans le cas où la zone *SubType* de la structure MQSBC est de valeur MQSUBTYPE\_PROXY, la zone *CorrelId* du descripteur de message ne doit pas être modifiée.

Aucun descripteur de message n'est transmis à l'exit si *ExitReason* est MQXR\_INIT ou MQXR\_TERM ; dans ces cas, *MsgDescPtr* est le pointeur null.

*MsgDescPtr* est une zone d'entrée de l'exit.

### **MsgHandle (MQHMSG)**

*MsgHandle* est le descripteur des propriétés de message. Utilisez uniquement *MsgHandle* avec les appels de fonction de propriété de message MQSETMP, MQINQMMP ou MQDLTMP pour utiliser les propriétés de message.

*MsgHandle* est une zone d'entrée de l'exit.

### **MsgInPtr (PMQVOID)**

*MsgInPtr* est l'adresse des données de message d'entrée. Le contenu de la mémoire tampon adressée par *MsgInPtr* peut être modifié par l'exit ; voir [MsgOutPtr](#) .

*MsgInPtr* est une zone d'entrée de l'exit.

### **MsgInLength (MQLONG)**

*MsgInLength* est la longueur en octets des données de message transmises à l'exit. L'adresse des données est indiquée par *MsgInPtr* .

*MsgInLength* est une zone d'entrée de l'exit.

### **MsgOutPtr (PMQVOID)**

*MsgOutPtr* est l'adresse d'une mémoire tampon contenant les données de message renvoyées par l'exit. A l'entrée de l'exit, *MsgOutPtr* est null. En retour de l'exit, si la valeur est toujours null, le gestionnaire de files d'attente envoie le message spécifié par *MsgInPtr* , avec la longueur indiquée par *MsgInLength* .

Si l'exit modifie les données de message, utilisez l'une des procédures suivantes:

- Si la longueur des données ne change pas, les données peuvent être modifiées dans la mémoire tampon adressée par *MsgInPtr* . Dans ce cas, ne modifiez pas *MsgOutPtr* et *MsgOutLength* .
- Si les données modifiées sont plus courtes que les données d'origine, les données peuvent être modifiées dans la mémoire tampon adressée par *MsgInPtr* . Dans ce cas, *MsgOutPtr* doit être défini sur l'adresse de la mémoire tampon du message d'entrée et *MsgOutLength* sur la nouvelle longueur des données du message.
- Si les données modifiées sont ou peuvent être plus longues que les données d'origine, l'exit doit obtenir une nouvelle mémoire tampon de messages. Copiez les données modifiées dans le fichier. Définissez *MsgOutPtr* sur l'adresse de la nouvelle mémoire tampon et définissez *MsgOutLength* sur la longueur des nouvelles données de message. L'exit est responsable de la libération de la mémoire tampon adressée par *MsgOutPtr* lors de l'appel suivant de l'exit.

**Remarque :** *MsgOutPtr* est toujours le pointeur null en entrée de l'exit, et non l'adresse d'une mémoire tampon de messages précédemment obtenue. Pour libérer la mémoire tampon précédemment obtenue, l'exit doit sauvegarder son adresse et sa longueur. Sauvegardez les informations dans *ExitUserArea* ou dans un bloc de contrôle dont l'adresse est sauvegardée dans *ExitUserArea* .

*MsgOutPtr* est une zone d'entrée-sortie de l'exit.

### **MsgOutLength (MQLONG)**

*MsgOutLength* est la longueur en octets des données de message renvoyées par l'exit. En entrée de l'exit, cette zone est toujours à zéro. Lors du retour de l'exit, cette zone est ignorée si *MsgOutPtr* est null. Pour plus d'informations sur la modification des données de message, voir [MsgOutPtr](#) .

*MsgOutLength* est une zone d'entrée-sortie de l'exit.

### **pEntryPoints (PMQIEP)**

*pEntryPoints* est l'adresse d'une structure MQIEP via laquelle les appels MQI et DCI peuvent être effectués.

## **Déclaration de langage C-MQPSXP**

```
typedef struct tagMQPSXP {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    ExitId;           /* Type of exit */
    MQLONG    ExitReason;       /* Reason for invoking exit */
    MQLONG    ExitResponse;     /* Response from exit */
    MQLONG    ExitResponse2;    /* Reserved */
    MQLONG    Feedback;        /* Feedback code */
    MQBYTE16  ExitUserArea;     /* Exit user area */
    MQCHAR32  ExitData;         /* Exit data */
    MQCHAR48  QMgrName;        /* Name of local queue manager */
    MQHCONN   Hconn;           /* Connection handle */
};
```

```

MQHMSG      MsgHandle;           /* Handle to message properties */
MQMD        MsgDescPtr;         /* Address of message descriptor */
MQVOID      MsgInPtr;          /* Address of input message data */
MQLONG      MsgInLength;       /* Length of input message data */
MQVOID      MsgOutPtr;         /* Address of output message data */
MQLONG      MsgOutLength;      /* Length of output message data */
/* Ver:1 */
MQIEP       pEntryPoints;      /* Address of the MQIEP structure */
/* Ver:2 */
} MQPSXP;

```

## MQPBC-Structure de données de contexte de publication

La structure MQPBC contient les informations contextuelles, relatives au diffuseur de publications de la publication, qui sont transmises à l'exit de publication.

Le [Tableau 819](#), à la page 1542 récapitule les zones de la structure:

Tableau 819. Zones dans MQPBC	
Zone	Description
<i>StrucID</i>	Identificateur de structure
<i>Version</i>	Numéro de version de structure
<i>PubTopicString</i>	Publier la chaîne de rubrique
<i>MsgDescPtr</i>	Adresse du descripteur de message (MQMD)

### Zones

#### **StrucID (MQCHAR4)**

*StrucID* est l'identificateur de structure. La valeur est la suivante:

##### **MQPBC\_STRUCID**

MQPBC\_STRUCID est l'identificateur de la structure de contexte de publication. Pour le langage de programmation C, la constante MQPBC\_STRUC\_ID\_ARRAY est également définie ; elle a la même valeur que MQPBC\_STRUC\_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

*StrucID* est une zone d'entrée de l'exit.

#### **Version (MQLONG)**

*Version* est le numéro de version de la structure. La valeur est la suivante:

##### **MQPBC\_VERSION\_1**

MQPBC\_VERSION\_1 est la structure des paramètres d'exit de publication version 1.

##### **MQPBC\_VERSION\_2**

MQPBC\_VERSION\_2 est la structure du paramètre d'exit de publication version 2. La constante MQPBC\_CURRENT\_VERSION est également définie avec la même valeur.

*Version* est une zone d'entrée de l'exit.

#### **PubTopicString (MQCHARV)**

*PubTopicString* est la chaîne de rubrique dans laquelle la publication est en cours.

*PubTopicString* est une zone d'entrée de l'exit.

#### **MsgDescPtr (MQMD)**

*MsgDescPtr* est l'adresse d'une copie du descripteur de message (MQMD) pour le message en cours de traitement.

*MsgDescPtr* est une zone d'entrée de l'exit.

## Déclaration de langage C-MQPBC

```
typedef struct tagMQPBC {
```

```

MQCHAR4   StrucId;           /* Structure identifier */
MQLONG    Version;          /* Structure version number */
MQCHARV   PubTopicString;  /* Publish topic string */
PMQMD     MsgDescPtr;      /* Address of message descriptor */
} MQPBC;

```

## MQSBC-Structure de données de contexte d'abonnement

La structure MQSBC contient les informations contextuelles relatives à l'abonné qui reçoit la publication et qui sont transmises à l'exit de publication.

Le Tableau 820, à la page 1543 récapitule les zones de la structure:

Tableau 820. Champs dans MQSBC	
Zone	Description
<u>StrucID</u>	Identificateur de structure
<u>Version</u>	Numéro de version de structure
<u>DestinationQMgrName</u>	Nom du gestionnaire de files d'attente de destination
<u>DestinationQName</u>	Nom de la file d'attente de destination
<u>SubType</u>	Type d'abonnement
<u>SubOptions</u>	Options d'abonnement
<u>ObjectName</u>	Nom d'objet
<u>ObjectString</u>	Chaîne d'objet
<u>SubTopicString</u>	Chaîne de rubrique d'abonnement
<u>SubName</u>	Nom de l'abonnement
<u>SubId</u>	Identificateur d'abonnement
<u>SelectionString</u>	Adresse de la chaîne de sélection
<u>SubLevel</u>	Niveau d'abonnement
<u>PSPProperties</u>	Propriétés de publication / abonnement

### Zones

#### **StrucID (MQCHAR4)**

Identificateur de structure. La valeur est la suivante:

##### **MQSBC\_STRUCID**

MQSBC\_STRUCID est l'identificateur de la structure des paramètres d'exit de publication. Pour le langage de programmation C, la constante MQSBC\_STRUC\_ID\_ARRAY est également définie ; MQSBC\_STRUC\_ID\_ARRAY a la même valeur que MQSBC\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

*StrucID* est une zone d'entrée de l'exit.

#### **Version (MQLONG)**

Numéro de version de la structure. La valeur est la suivante:

##### **MQSBC\_VERSION\_1**

Structure des paramètres d'exit de publication de la version 1. La constante MQSBC\_CURRENT\_VERSION est également définie avec la même valeur.

*Version* est une zone d'entrée de l'exit.

**DestinationQMgrName (MQCHAR48)**

*DestinationQMgrName* est le nom du gestionnaire de files d'attente auquel le message est envoyé. Le nom est rempli avec des blancs sur toute la longueur de la zone. Le nom peut être modifié par l'exit. La longueur de cette zone est indiquée par MQ\_Q\_MGR\_NAME\_LENGTH.

*DestinationQMgrName* est une zone d'entrée-sortie pour l'exit ; voir remarque.

**DestinationQName (MQCHAR48)**

*DestinationQName* est le nom de la file d'attente à laquelle le message est envoyé. Le nom est rempli avec des blancs sur toute la longueur de la zone. Le nom peut être modifié par l'exit. La longueur de cette zone est indiquée par MQ\_Q\_NAME\_LENGTH.

*DestinationQName* est une zone d'entrée-sortie pour l'exit ; voir remarque.

**SubType (MQLONG)**

*SubType* indique comment l'abonnement a été créé. Les valeurs valides sont MQSUBTYPE\_API, MQSUBTYPE\_ADMIN et MQSUBTYPE\_PROXY ; voir Inquire Subscription Status (Response).

*SubType* est une zone d'entrée de l'exit.

**SubOptions (MQLONG)**

*SubOptions* sont les options d'abonnement ; voir «Options (MQLONG) pour MQSD», à la page 599 pour une description des valeurs que cette zone peut prendre.

*SubOptions* est une zone d'entrée de l'exit.

**ObjectName (MQCHAR48)**

*ObjectName* est le nom de l'objet de rubrique tel qu'il est défini sur le gestionnaire de files d'attente local. La longueur de cette zone est indiquée par MQ\_TOPIC\_NAME\_LENGTH. Le nom de l'objet est le nom de l'objet de rubrique d'administration que le gestionnaire de files d'attente a associé à la chaîne de rubrique. Même si l'abonné a fourni un objet de rubrique dans le cadre de l'abonnement, *ObjectName* peut être un objet de rubrique différent. L'association d'un objet de rubrique à un abonnement dépend de la résolution complète de *SubTopicString*.

*ObjectName* est une zone d'entrée de l'exit.

**ObjectString (MQCHARV)**

*ObjectString* est la chaîne de rubrique complète de la publication à laquelle vous avez souscrit. Tous les caractères génériques de la chaîne d'abonnement d'origine sont résolus. Il est différent de la zone MQSD subscription *ObjectString* décrite dans «ObjectString (MQCHARV) pour MQSD», à la page 609, qui peut contenir des caractères génériques et exclut tout nom d'objet fourni par l'abonné.

*ObjectString* est une zone d'entrée de l'exit.

**SubTopicString (MQCHARV)**

*SubTopicString* est la chaîne de rubrique complète fournie par l'abonné. *SubTopicString* est la combinaison de la chaîne de rubrique définie dans un objet de rubrique et d'une chaîne de rubrique. Un abonné doit fournir un objet de rubrique, une chaîne de rubrique ou les deux. Si l'abonné fournit une chaîne de rubrique, elle peut contenir des caractères génériques.

*SubTopicString* est une zone d'entrée de l'exit.

**SubName (MQCHARV)**

*SubName* est le nom d'abonnement fourni par l'abonné ou est un nom généré.

*SubName* est une zone d'entrée de l'exit.

**SubId (MQBYTE 24)**

*SubId* est l'identificateur d'abonnement interne unique.

*SubId* est une zone d'entrée de l'exit.

**SelectionString (MQCHARV)**

*SelectionString* correspond aux critères de sélection utilisés lors de l'abonnement aux messages d'une rubrique ; voir Sélecteurs.

*SelectionString* est une zone d'entrée de l'exit.



### SubLevel (MQLONG)

*SubLevel* est le niveau d'interception associé à l'abonnement ; voir «[SubLevel \(MQLONG\) pour MQSD](#)», à la page 613 pour plus de détails.

*SubLevel* est une zone d'entrée de l'exit.

### PSPProperties (MQLONG)

*PSPProperties* sont les propriétés de publication / abonnement. Ils spécifient comment les propriétés de message liées à la publication / abonnement sont ajoutées aux messages envoyés à cet abonnement. Les valeurs possibles sont MQPSPROP\_NONE, MQPSPROP\_COMPAT, MQPSPROP\_RFH2, MQPSPROP\_MSGPROP. Pour obtenir une description de ces valeurs, voir [Paramètres facultatifs \(Modifier, Copier et Créer un abonnement\)](#) .

*PSPProperties* est une zone d'entrée de l'exit.

**Remarque :** Les vérifications d'autorisation ne sont effectuées que sur les valeurs d'origine de *DestinationQMgrName* et *DestinationQName* avant qu'elles ne soient transmises à l'exit de publication. Aucune nouvelle vérification d'autorisation n'est effectuée lorsque l'exit modifie la file d'attente de destination, en modifiant *DestinationQMgrName* ou *DestinationQName*.

## Déclaration de langage C-MQSBC

```
typedef struct tagMQSBC {
    MQCHAR4   StructId;           /* Structure identifier */
    MQLONG    Version;           /* Structure version number */
    MQCHAR48  DestinationQMgrName; /* Destination queue manager */
    MQCHAR48  DestinationQName;  /* Destination queue name */
    MQLONG    SubType;           /* Type of subscription */
    MQLONG    SubOptions;        /* Subscription options */
    MQCHAR48  ObjectName;        /* Object name */
    MQCHARV   ObjectString;      /* Object string */
    MQCHARV   SubTopicString;    /* Subscription topic string */
    MQCHARV   SubName;           /* Subscription name */
    MQBYTE24  SubId;             /* Subscription identifier */
    MQCHARV   SelectionString;   /* Subscription selection string */
    MQLONG    SubLevel;         /* Subscription level */
    MQLONG    PSPProperties;     /* Publish/subscribe properties */
} MQSBC;
```

## Structures de données et appels d'exit de canal

Cette collection de rubriques fournit des informations de référence sur les appels IBM MQ spéciaux et les structures de données que vous pouvez utiliser lorsque vous écrivez des programmes d'exit de canal.

Ces informations sont des informations d'interface de programmation sensibles au produit. Vous pouvez écrire des exits utilisateur IBM MQ dans les langages de programmation suivants:

Tableau 821. Exits utilisateur IBM MQ : plateformes et langages de programmation	
Plateforme	Langages de programmation
IBM MQ for z/OS	Assembleur et C (qui doivent être conformes à l'environnement de programmation système C pour les exits système, décrits dans le document <i>z/OS C/C++ Programming Guide</i> .)
IBM MQ for IBM i	ILE C, ILE COBOL et ILE RPG
Toutes les autres plateformes IBM MQ	C

Vous pouvez également écrire des exits utilisateur dans Java à utiliser uniquement avec les applications Java et JMS . Pour plus d'informations sur la création et l'utilisation d'exits de canal avec IBM MQ classes for Java, voir [Utilisation d'exits de canal dans IBM MQ classes for Java](#) et pour IBM MQ classes for JMS, voir [Utilisation d'exits de canal avec IBM MQ classes for JMS](#).

Vous ne pouvez pas écrire d'exits utilisateur IBM MQ dans TAL ou Visual Basic. Toutefois, une déclaration pour la structure MQCD est fournie dans Visual Basic pour une utilisation sur l'appel MQCONNX à partir d'un programme IBM MQ MQI client .

Dans un certain nombre de cas dans les descriptions qui suivent, les paramètres sont des tableaux ou des chaînes de caractères dont la taille n'est pas fixe. Pour ces paramètres, une minuscule "n" est utilisée pour représenter une constante numérique. Lorsque la déclaration de ce paramètre est codée, la valeur "n" doit être remplacée par la valeur numérique requise. Pour plus d'informations sur les conventions utilisées dans ces descriptions, voir [«Types de données élémentaires»](#), à la page 238.

## Fichiers de définition de données

Les fichiers de définition de données sont fournis avec IBM MQ pour chacun des langages de programmation pris en charge. Pour plus de détails sur ces fichiers, voir [Copy, header, include et module files](#).

## MQ\_CHANNEL\_EXIT-Exit de canal

L'appel MQ\_CHANNEL\_EXIT décrit les paramètres qui sont transmis à chacun des exits de canal appelés par l'agent MCA.

Aucun point d'entrée appelé MQ\_CHANNEL\_EXIT n'est fourni par le gestionnaire de files d'attente ; le nom MQ\_CHANNEL\_EXIT n'a pas de signification particulière car les noms des exits de canal sont fournis dans la définition de canal MQCD.

Il existe cinq types d'exit de canal:

- Exit de sécurité de canal
- Exit de message de canal
- Exit d'émission de canal
- Exit de réception de canal
- Message du canal-exit de relance

Les paramètres sont similaires pour chaque type de sortie, et la description donnée ici s'applique à tous, sauf mention spécifique.

## Syntaxe

**MQ\_CHANNEL\_EXIT** (*ChannelExitParms*, *ChannelDefinition*, *DataLength*, *AgentBufferLength*, *AgentBuffer*, *ExitBufferLength*, *ExitBufferAddr*)

## Paramètres

L'appel MQ\_CHANNEL\_EXIT possède les paramètres suivants.

### ChannelExitParamètres (MQCXP)-entrée / sortie

Bloc de paramètres d'exit de canal.

Cette structure contient des informations supplémentaires relatives à l'appel de l'exit. L'exit définit les informations de cette structure pour indiquer comment l'agent MCA se déroule.

### ChannelDefinition (MQCD)-entrée/sortie

Définition de canal.

Cette structure contient des paramètres définis par l'administrateur pour contrôler le comportement du canal.

### DataLength (MQLONG)-entrée/sortie

Longueur des données.

Les données dépendent du type d'exit:

- Pour un exit de sécurité de canal, lorsque l'exit est appelé, ce paramètre contient la longueur de tout message de sécurité dans la zone *AgentBuffer*, si *ExitReason* est MQXR\_SEC\_MSG. Il est égal à zéro s'il n'y a pas de message. L'exit doit définir cette zone sur la longueur de tout message de sécurité à envoyer à son partenaire s'il définit *ExitResponse* sur MQXCC\_SEND\_SEC\_MSG ou MQXCC\_SEND\_AND\_REQUEST\_SEC\_MSG. Les données de message se trouvent dans *AgentBuffer* ou *ExitBufferAddr*.

Le contenu des messages de sécurité est la seule responsabilité des exits de sécurité.

- Pour un exit de message de canal, lorsque l'exit est appelé, ce paramètre contient la longueur du message (y compris l'en-tête de la file d'attente de transmission). L'exit doit définir cette zone sur la longueur du message dans *AgentBuffer* ou *ExitBufferAddr* qui doit être traitée. Elle doit être supérieure ou égale à la longueur de l'en-tête de la file d'attente de transmission (MQXQH).
- Pour un exit d'émission ou de réception de canal, lorsque l'exit est appelé, ce paramètre contient la longueur de la transmission. L'exit doit définir cette zone sur la longueur de la transmission dans *AgentBuffer* ou *ExitBufferAddr* qui doit se poursuivre.

Si un exit de sécurité envoie un message et qu'il n'y a pas d'exit de sécurité à l'autre extrémité du canal, ou que l'autre extrémité définit un *ExitResponse* de MQXCC\_OK, l'exit initiateur est à nouveau appelé avec MQXR\_SEC\_MSG et une réponse nulle (*DataLength* = 0).

### **AgentBufferLength (MQLONG)-entrée**

Longueur de la mémoire tampon de l'agent.

Ce paramètre peut être supérieur à *DataLength* lors de l'appel.

Pour les exits de message de canal, d'envoi et de réception, tout espace inutilisé lors de l'appel peut être utilisé par l'exit pour développer les données en place. Dans ce cas, le paramètre **DataLength** doit être défini de manière appropriée par l'exit.

Dans le langage de programmation C, ce paramètre est transmis par adresse.

### **AgentBuffer (MQBYTE x AgentBufferLength)-entrée/sortie**

Mémoire tampon de l'agent.

Le contenu de ce paramètre dépend du type d'exit:

- Pour un exit de sécurité de canal, lors de l'appel de l'exit, il contient un message de sécurité si *ExitReason* est MQXR\_SEC\_MSG. Pour renvoyer un message de sécurité, l'exit peut utiliser cette mémoire tampon ou sa propre mémoire tampon (*ExitBufferAddr*).
- Pour un exit de message de canal, lors de l'appel de l'exit, ce paramètre contient:
  - L'en-tête de file d'attente de transmission (MQXQH), qui inclut le descripteur de message (qui contient lui-même les informations de contexte du message), immédiatement suivi par
  - Données de message

Si le message doit continuer, l'exit peut effectuer l'une des opérations suivantes:

- Ne pas toucher au contenu de la mémoire tampon
- Modifier le contenu en place (renvoi de la nouvelle longueur des données dans *DataLength*; ne doit pas être supérieur à *AgentBufferLength*)
- Copiez le contenu dans le *ExitBufferAddr*, en y apportant les modifications requises

Les modifications apportées par l'exit à l'en-tête de la file d'attente de transmission ne sont pas vérifiées; toutefois, des modifications erronées peuvent signifier que le message ne peut pas être placé à la destination.

- Pour un exit d'émission ou de réception de canal, lors de l'appel de l'exit, celui-ci contient les données de transmission. L'exit peut effectuer l'une des opérations suivantes:
  - Ne pas toucher au contenu de la mémoire tampon
  - Modifier le contenu en place (renvoi de la nouvelle longueur des données dans *DataLength*; ne doit pas être supérieur à *AgentBufferLength*)

- Copiez le contenu dans le *ExitBufferAddr*, en y apportant les modifications requises

Les 8 premiers octets des données ne doivent pas être modifiés par l'exit.

### ExitBufferLength (MQLONG)-entrée/sortie

Longueur de la mémoire tampon de sortie.

Lors du premier appel de l'exit, ce paramètre est défini sur zéro. Ensuite, toute valeur renvoyée par l'exit, à chaque appel, est présentée à l'exit lors de son prochain appel. La valeur n'est pas utilisée par l'agent MCA.

**Remarque :** Ce paramètre ne doit pas être utilisé par des exits écrits dans des langages de programmation qui ne prennent pas en charge le type de données de pointeur.

### ExitBufferAddr (MQPTR)-entrée/sortie

Adresse de la mémoire tampon de sortie.

Ce paramètre est un pointeur vers l'adresse d'une mémoire tampon gérée par l'exit, où il peut choisir de renvoyer des données de message ou de transmission (en fonction du type d'exit) à l'agent si la mémoire tampon de l'agent est ou peut ne pas être suffisamment grande, ou s'il est plus pratique pour l'exit de le faire.

Lors du premier appel de l'exit, l'adresse transmise à l'exit est null. Par la suite, toute adresse transmise par l'exit, à chaque appel, est présentée à l'exit lors de son prochain appel.

Si ExitBufferAddr est null, les données utilisées sont extraites du paramètre AgentBuffer .

Si ExitBufferAddr n'est pas null, les données utilisées sont extraites de la mémoire tampon vers laquelle pointe le paramètre ExitBufferAddr.

**Remarque :** Ce paramètre ne doit pas être utilisé par des exits écrits dans des langages de programmation qui ne prennent pas en charge le type de données de pointeur.

## Appel C

```
exitname (&ChannelExitParms, &ChannelDefinition,
&DataLength, &AgentBufferLength, AgentBuffer,
&ExitBufferLength, &ExitBufferAddr);
```

Les paramètres transmis à l'exit sont déclarés comme suit:

```
MQCXP  ChannelExitParms; /* Channel exit parameter block */
MQCD   ChannelDefinition; /* Channel definition */
MQLONG DataLength; /* Length of data */
MQLONG AgentBufferLength; /* Length of agent buffer */
MQBYTE AgentBuffer[n]; /* Agent buffer */
MQLONG ExitBufferLength; /* Length of exit buffer */
MQPTR  ExitBufferAddr; /* Address of exit buffer */
```

## Appel COBOL

```
CALL 'exitname' USING CHANNELEXITPARMS, CHANNELDEFINITION,
DATALENGTH, AGENTBUFFERLENGTH, AGENTBUFFER,
EXITBUFFERLENGTH, EXITBUFFERADDR.
```

Les paramètres transmis à l'exit sont déclarés comme suit:

```
** Channel exit parameter block
01 CHANNELEXITPARMS.
   COPY CMQCXPV.
** Channel definition
01 CHANNELDEFINITION.
   COPY CMQCDV.
** Length of data
```

```

01 DATALENGTH          PIC S9(9) BINARY.
** Length of agent buffer
01 AGENTBUFFERLENGTH   PIC S9(9) BINARY.
** Agent buffer
01 AGENTBUFFER          PIC X(n).
** Length of exit buffer
01 EXITBUFFERLENGTH    PIC S9(9) BINARY.
** Address of exit buffer
01 EXITBUFFERADDR      POINTER.

```

## Appel RPG (ILE)

```

C*.1.....2.....3.....4.....5.....6.....7..
C          CALLP      exitname(MQCXP : MQCD : DATLEN :
C                               ABUFL : ABUF : EBUFL :
C                               EBUF)

```

La définition de prototype pour l'appel est la suivante:

```

D*.1.....2.....3.....4.....5.....6.....7..
Dexitname          PR          EXTPROC('exitname')
D* Channel exit parameter block
D MQCXP              160A
D* Channel definition
D MQCD              1328A
D* Length of data
D DATLEN            10I 0
D* Length of agent buffer
D ABUFL             10I 0
D* Agent buffer
D ABUF              *      VALUE
D* Length of exit buffer
D EBUFL            10I 0
D* Address of exit buffer
D EBUF             *

```

## Appel de l'assembleur System/390

```

CALL EXITNAME, (CHANNELEXITPARMS, CHANNELDEFINITION, DATALENGTH, X
               AGENTBUFFERLENGTH, AGENTBUFFER, EXITBUFFERLENGTH, X
               EXITBUFFERADDR)

```

Les paramètres transmis à l'exit sont déclarés comme suit:

CHANNELEXITPARMS	CMQCXPA	,	Channel exit parameter block
CHANNELDEFINITION	CMQCDA	,	Channel definition
DATALENGTH	DS	F	Length of data
AGENTBUFFERLENGTH	DS	F	Length of agent buffer
AGENTBUFFER	DS	CL(n)	Agent buffer
EXITBUFFERLENGTH	DS	F	Length of exit buffer
EXITBUFFERADDR	DS	F	Address of exit buffer

## Remarques sur l'utilisation

1. La fonction exécutée par l'exit de canal est définie par le fournisseur de l'exit. L'exit, cependant, doit respecter les règles définies ici et dans le bloc de contrôle associé, le MQCXP.
2. Le paramètre **ChannelDefinition** transmis à l'exit de canal peut être l'une des versions suivantes. Pour plus d'informations, voir la zone *Version* dans la structure MQCD.
3. Si l'exit de canal reçoit une structure MQCD avec la zone *Version* définie sur une valeur supérieure à MQCD\_VERSION\_1, l'exit doit utiliser la zone *ConnectionName* dans MQCD, de préférence à la zone *ShortConnectionName*.
4. En général, les exits de canal sont autorisés à modifier la longueur des données de message. Cela peut être dû à l'ajout de données au message par l'exit, à la suppression de données du message ou

à la compression ou au chiffrement du message. Toutefois, des restrictions spéciales s'appliquent si le message est un segment qui ne contient qu'une partie d'un message logique. En particulier, il ne doit pas y avoir de changement net de la longueur du message à la suite des actions des sorties d'émission et de réception complémentaires.

Par exemple, il est permis à un exit d'émission de raccourcir le message en le compressant, mais l'exit de réception complémentaire doit restaurer la longueur d'origine du message en le décompressant, de sorte qu'il n'y ait pas de changement net de la longueur du message.

Cette restriction est due au fait que la modification de la longueur d'un segment provoquerait une inexactitude des décalages des segments ultérieurs du message, ce qui empêcherait le gestionnaire de files d'attente de reconnaître que les segments formaient un message logique complet.

## **MQ\_CHANNEL\_AUTO\_DEF\_EXIT-Exit de définition automatique de canal**

L'appel MQ\_CHANNEL\_AUTO\_DEF\_EXIT décrit les paramètres transmis à l'exit de définition automatique de canal appelé par l'agent MCA.

Aucun point d'entrée appelé MQ\_CHANNEL\_AUTO\_DEF\_EXIT n'est fourni par le gestionnaire de files d'attente ; le nom MQ\_CHANNEL\_AUTO\_DEF\_EXIT n'a pas de signification particulière car les noms des exits de définition automatique sont fournis dans le gestionnaire de files d'attente.

### **Syntaxe**

**MQ\_CHANNEL\_AUTO\_DEF\_EXIT** (*ChannelExitParms*, *ChannelDefinition*)

### **Paramètres**

L'appel MQ\_CHANNEL\_AUTO\_DEF\_EXIT possède les paramètres suivants.

#### **ChannelExitParamètres (MQCXP)-entrée / sortie**

Bloc de paramètres d'exit de canal.

Cette structure contient des informations supplémentaires relatives à l'appel de l'exit. L'exit définit les informations de cette structure pour indiquer comment l'agent MCA se déroule.

#### **ChannelDefinition (MQCD)-entrée/sortie**

Définition de canal.

Cette structure contient des paramètres définis par l'administrateur pour contrôler le comportement des canaux créés automatiquement. L'exit définit des informations dans cette structure pour modifier le comportement par défaut défini par l'administrateur.

Les zones MQCD répertoriées ne doivent pas être modifiées par l'exit:

- *ChannelName*
- *ChannelType*
- *StrucLength*
- *Version*

Si d'autres zones sont modifiées, la valeur définie par l'exit doit être valide. Si la valeur n'est pas valide, un message d'erreur est consigné dans le fichier journal des erreurs ou affiché sur la console (selon l'environnement).



**Avertissement :** Les canaux définis automatiquement créés par un exit de définition automatique de canal (CHAD) ne peuvent pas définir le libellé de certificat car l'établissement de liaison TLS a eu lieu au moment de la création du canal. La définition du libellé de certificat dans un exit CHAD pour les canaux entrants n'a aucun effet.

## Appel C

```
exitname (&ChannelExitParms, &ChannelDefinition);
```

Les paramètres transmis à l'exit sont déclarés comme suit:

```
MQCXP ChannelExitParms; /* Channel exit parameter block */
MQCD ChannelDefinition; /* Channel definition */
```

## Appel COBOL

```
CALL 'exitname' USING CHANNELEXITPARMS, CHANNELDEFINITION.
```

Les paramètres transmis à l'exit sont déclarés comme suit:

```
** Channel exit parameter block
01 CHANNELEXITPARMS.
   COPY CMQCXPV.
** Channel definition
01 CHANNELDEFINITION.
   COPY CMQCDV.
```

## Appel RPG (ILE)

```
C*.1.....2.....3.....4.....5.....6.....7..
C CALLP exitname(MQCXP : MQCD)
```

La définition de prototype pour l'appel est la suivante:

```
D*.1.....2.....3.....4.....5.....6.....7..
Dexitname PR EXTPROC('exitname')
D* Channel exit parameter block
D MQCXP 160A
D* Channel definition
D MQCD 1328A
```

## Appel de l'assembleur System/390

```
CALL EXITNAME, (CHANNELEXITPARMS, CHANNELDEFINITION)
```

Les paramètres transmis à l'exit sont déclarés comme suit:

```
CHANNELEXITPARMS CMQCXPA , Channel exit parameter block
CHANNELDEFINITION CMQCDA , Channel definition
```

## Remarques sur l'utilisation


1. La fonction exécutée par l'exit de canal est définie par le fournisseur de l'exit. L'exit, cependant, doit respecter les règles définies ici et dans le bloc de contrôle associé, le MQCXP.
2. Le paramètre **ChannelExitParms** transmis à l'exit de définition automatique de canal est une structure MQCXP. La version de MQCXP transmise dépend de l'environnement dans lequel l'exit est exécuté ; voir la description de la zone *Version* dans [«MQCXP-Paramètre d'exit de canal»](#), à la page 1595 pour plus de détails.
3. Le paramètre **ChannelDefinition** transmis à l'exit de définition automatique de canal est une structure MQCD. La version de MQCD transmise dépend de l'environnement dans lequel l'exit est en

cours d'exécution ; voir la description de la zone *Version* dans «MQCD-Définition de canal», à la page 1553 pour plus de détails.

## MQXWAIT-Attendre l'exit

L'appel MQXWAIT attend qu'un événement se produise. Il ne peut être utilisé qu'à partir d'un exit de canal sous z/OS.

L'utilisation de MQXWAIT permet d'éviter les problèmes de performances qui pourraient se produire si un exit de canal effectue une action qui provoque une attente. L'événement MQXWAIT en attente est signalé par un bloc de contrôle d'événement (ECB) MVS. L'ECB est décrit dans la description du bloc de contrôle MQXWD.

 Pour plus d'informations sur l'utilisation de MQXWAIT et l'écriture de programmes d'exit de canal, voir [Ecriture de programmes d'exit de canal sur z/OS](#)

### Syntaxe

**MQXWAIT** (*Hconn*, *WaitDesc*, *CompCode*, *Reason*)

### Paramètres

L'appel MQXWAIT comporte les paramètres suivants.

#### Hconn (MQHCONN)-entrée

Descripteur de connexion.

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN précédent émis lors de l'appel identique ou antérieur de l'exit.

#### WaitDesc (MQXWD)-entrée/sortie

Descripteur d'attente.

Ce paramètre décrit l'événement à attendre. Pour plus de détails sur les zones de cette structure, voir «MQXWD-Descripteur d'attente d'exit», à la page 1610 .

#### CompCode (MQLONG)-sortie

Code achèvement.

Il s'agit de l'un des codes suivants:

##### **MQCC\_OK**

Achèvement réussi.

##### **MQCC\_FAILED**

Echec de l'appel.

#### Cause (MQLONG)-sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

##### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

##### **MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') Adaptateur non disponible.

##### **MQRC\_OPTIONS\_ERROR**

(2046, X'7FE') Options incorrectes ou incohérentes.

##### **MQRC\_XWAIT\_ANNULE**

(2107, X'83B') Appel MQXWAIT annulé.



## MQRC\_XWAIT\_ERREUR

(2108, X'83C') Appel de l'appel MQXWAIT non valide.

### Appel C

```
MQXWAIT (Hconn, &WaitDesc, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;      /* Connection handle */
MQXWD    WaitDesc;  /* Wait descriptor */
MQLONG   CompCode; /* Completion code */
MQLONG   Reason;   /* Reason code qualifying CompCode */
```

### Appel de l'assembleur System/390

```
CALL MQXWAIT, (HCONN, WAITDESC, COMPCODE, REASON)
```

Déclarez les paramètres comme suit :

```
HCONN      DS      F  Connection handle
WAITDESC   CMQXWDA ,  Wait descriptor
COMPCODE   DS      F  Completion code
REASON     DS      F  Reason code qualifying COMPCODE
```

## MQCD-Définition de canal

La structure MQCD contient les paramètres qui contrôlent l'exécution d'un canal. Il est transmis à chaque exit de canal appelé à partir d'un agent MCA (Message Channel Agent).

Pour plus d'informations sur les exits de canal, voir «MQ\_CHANNEL\_EXIT-Exit de canal», à la page 1546. La description de cette rubrique concerne à la fois les canaux de transmission de messages et les canaux MQI.

### Zones de nom de sortie

Lorsqu'un exit est appelé, la zone appropriée de *SecurityExit*, *MsgExit*, *SendExit*, *ReceiveExit* et *MsgRetryExit* contient le nom de l'exit actuellement appelé. La signification du nom dans ces zones dépend de l'environnement dans lequel l'agent MCA s'exécute. Sauf indication contraire, le nom est aligné à gauche dans la zone, sans blancs imbriqués ; le nom est complété par des blancs à la longueur de la zone. Dans les descriptions qui suivent, les crochets ([ ]) indiquent des informations facultatives:

#### AIX and Linux

Le nom de l'exit est le nom d'un module ou d'une bibliothèque chargeable dynamiquement, suffixé avec le nom d'une fonction résidant dans cette bibliothèque. Le nom de la fonction doit être placé entre parenthèses. Le nom de la bibliothèque peut éventuellement être précédé d'un chemin de répertoire:

```
[ path ] library ( function )
```

Le nom est limité à un maximum de 128 caractères.

#### z/OS

Le nom d'exit est le nom d'un module de chargement qui est valide pour la spécification sur le paramètre EP de la macro LINK ou LOAD. Le nom est limité à un maximum de huit caractères.

## Windows

Le nom d'exit est le nom d'une bibliothèque de liens dynamiques, suffixé avec le nom d'une fonction résidant dans cette bibliothèque. Le nom de la fonction doit être placé entre parenthèses. Le nom de la bibliothèque peut éventuellement être précédé d'un chemin de répertoire et d'une unité:

```
[d:][ path ] library ( fonction )
```

Le nom est limité à un maximum de 128 caractères.

## IBM i

Le nom de l'exit est un nom de programme de 10 octets suivi d'un nom de bibliothèque de 10 octets. Si la longueur des noms est inférieure à 10 octets, chaque nom est complété par des blancs pour en faire 10 octets. Le nom de la bibliothèque peut être \*LIBL sauf lors de l'appel d'un exit de définition automatique de canal, auquel cas un nom qualifié complet est requis.

## Modification des zones MQCD dans un exit de canal

Un exit de canal peut modifier les zones de la base de données MQCD. La valeur modifiée reste dans le MQCD et est transmise à tous les exits restants d'une chaîne d'exit et à toute conversation partageant l'instance de canal. Le MQCD modifié est également utilisé par l'agent MCA pour son traitement normal pendant la durée de vie continue du canal.

Les zones MQCD suivantes ne doivent pas être modifiées par l'exit:

- ChannelName
- ChannelType
- StrucLength
- Version

### Référence associée

«Zones», à la page 1554

Cette rubrique répertorie toutes les zones de la structure MQCD et décrit chaque zone.

«Déclaration C», à la page 1582

Cette déclaration est la déclaration C pour la structure MQCD.

«Déclaration COBOL», à la page 1584

Cette déclaration est la déclaration COBOL pour la structure MQCD.

«Déclaration RPG (ILE)», à la page 1586

Cette déclaration est la déclaration RPG pour la structure MQCD.

«Déclaration de l'assembleur System/390», à la page 1589

Cette déclaration est la déclaration d'assembleur System/390 pour la structure MQCD.

«Déclaration Visual Basic», à la page 1590

Cette déclaration est la déclaration Visual Basic de la structure MQCD.

«Modification des zones MQCD dans un exit de canal», à la page 1592

Un exit de canal peut modifier les zones de la base de données MQCD. Toutefois, ces modifications ne sont généralement pas prises en compte, sauf dans les circonstances indiquées.

## Zones

Cette rubrique répertorie toutes les zones de la structure MQCD et décrit chaque zone.

### *BatchDataLimit (MQLONG)*

Cette zone indique la limite, en kilooctets, de la quantité de données pouvant être envoyée via un canal avant de prendre un point de synchronisation.

Un point de synchronisation est défini après le passage dans le canal du message qui a entraîné l'atteinte de la limite.

Le traitement du lot est terminé lorsque l'une des conditions suivantes est satisfaite :

- **BatchSize** messages ont été envoyés.
- **BatchDataLimit** octets ont été envoyés.
- La file d'attente de transmission est vide et **BatchInterval** est dépassé.

La valeur doit être comprise entre 0 et 999999. La valeur par défaut est 5000.

La valeur zéro dans cet attribut signifie qu'aucune limite de données n'est appliquée aux lots sur ce canal.

Ce paramètre s'applique uniquement aux canaux dont le *ChannelType* est MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_CLUSRCVR ou MQCHT\_CLUSSDR.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_11.

#### *BatchHeartbeat (MQLONG)*

Cette zone indique l'intervalle de temps utilisé pour déclencher un signal de présence de lot pour le canal.

La pulsation par lots permet aux canaux émetteurs de déterminer si l'instance de canal distant est toujours active avant de passer en attente de validation. Un signal de présence par lots se produit si un canal émetteur n'a pas communiqué avec l'instance de canal distant dans l'intervalle de temps spécifié.

La valeur est comprise entre 0 et 999 999 ; les unités sont des millisecondes. La valeur zéro indique que le signal de présence du lot n'est pas activé.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_CLUSSDR ou MQCHT\_CLUSRCVR.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_7.

#### *BatchInterval (MQLONG)*

Cette zone indique la durée approximative, en millisecondes, pendant laquelle un canal maintient un lot ouvert, si moins de *BatchSize* messages ont été transmis dans le lot en cours.

Si *BatchInterval* est supérieur à zéro, le lot est arrêté par l'un des événements suivants qui se produit en premier:

- Des messages *BatchSize* ont été envoyés ou
- *BatchInterval* millisecondes se sont écoulées depuis le démarrage du lot.

Si *BatchInterval* est égal à zéro, le lot est arrêté en premier lieu par l'un des événements suivants:

- Des messages *BatchSize* ont été envoyés ou
- la file d'attente de transmission devient vide.

*BatchInterval* doit être compris entre zéro et 999 999 999.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_CLUSSDR ou MQCHT\_CLUSRCVR.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente lorsque *Version* est inférieur à MQCD\_VERSION\_4.

#### *BatchSize (MQLONG)*

Cette zone indique le nombre maximal de messages pouvant être envoyés via un canal avant la synchronisation du canal.

Cette zone n'est pas pertinente pour les canaux dont le *ChannelType* est MQCHT\_SVRCONN ou MQCHT\_CLNTCONN.

#### *CertificateLabel (MQCHAR64)*

Cette zone fournit des détails sur le label de certificat utilisé.

IBM MQ initialise la valeur par défaut de la zone *CertificateLabel* sous la forme de blancs.

Cette valeur est interprétée lors de l'exécution comme la valeur par défaut et est compatible en amont.

Par exemple, si vous spécifiez une version de MQCD inférieure à 11 ou que vous utilisez la valeur par défaut des blancs pour la zone *CertificateLabel*, cette zone est ignorée.

La longueur de cette zone est indiquée par MQ\_CERT\_LABEL\_LENGTH.

#### *ChannelMonitoring (MQLONG)*

Cette zone indique le niveau en cours de la collecte de données de surveillance pour le canal.

Cette zone n'est pas pertinente pour les canaux dont le ChannelType est MQCHT\_CLNTCONN.

Ses valeurs sont l'une des suivantes :

- MQMON\_OFF
- MQMON\_FAIBLE
- MQMON\_MEDIUM
- MQMON\_ELEVE

Il s'agit d'une zone d'entrée de l'exit. Elle n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_8.

#### *ChannelName (MQCHAR20)*

Cette zone indique le nom de la définition de canal.

Il doit exister une définition de canal du même nom sur la machine distante pour pouvoir communiquer.

Le nom ne doit utiliser que les caractères suivants:

- Majuscules A-Z
- Minuscules a-z
- Chiffres 0-9
- le point (.)
- la barre oblique (/)
- le caractère de soulignement (\_)
- le symbole de pourcentage (%)

et être rempli à droite avec des blancs. Les espaces de début ou imbriqués ne sont pas autorisés.

La longueur de cette zone est donnée par MQ\_CHANNEL\_NAME\_LENGTH.

#### *ChannelStatistics (MQLONG)*

Cette zone indique le niveau actuel de collecte de données statistiques pour le canal.

Cette zone n'est pas pertinente pour les canaux dont le ChannelType est MQCHT\_CLNTCONN ou MQCHT\_SVRCONN.

Ses valeurs sont l'une des suivantes :

- MQMON\_OFF
- MQMON\_FAIBLE
- MQMON\_MEDIUM
- MQMON\_ELEVE

Il s'agit d'une zone d'entrée de l'exit. Elle n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_8.

#### *ChannelType (MQLONG)*

Cette zone indique le type de canal.

Ses valeurs sont l'une des suivantes :

**EXPÉDITEUR\_MQCH**

Expéditeur.

**SERVEUR\_MQ**

Serveur.

**MQCHT\_RECEIVER**

Récepteur.

**DEMANDE\_MQCHT\_DEMANDEUR**

Demandeur.

**MQCHT\_CLNTCONN**

Connexion client.

**MQCHT\_SVRCONN**

Connexion serveur (à utiliser par les clients).

**MQCHT\_CLUSSDR**

Emetteur de cluster.

**MQCHT\_CLUSRCVR**

Récepteur de cluster.

*Pondération ClientChannel(MQLONG)*

Cette zone indique une pondération qui influence la définition de canal de connexion client utilisée.

L'attribut de pondération ClientChannel est utilisé pour que les définitions de canal du client puissent être sélectionnées de manière aléatoire en fonction de leur pondération lorsque plusieurs définitions appropriées sont disponibles. Lorsqu'un client émet une demande de connexion MQCONN à un groupe de gestionnaires de files d'attente en spécifiant un nom de gestionnaire de files d'attente commençant par un astérisque et que plusieurs définitions de canal appropriées sont disponibles dans la table de définition de canal du client (CCDT), la définition à utiliser est sélectionnée de manière aléatoire en fonction de la pondération, avec les définitions ClientChannel(0) applicables sélectionnées en premier par ordre alphabétique.

Spécifiez une valeur comprise entre 0 et 99. La valeur par défaut est 0.

Elle indique qu'aucun équilibrage de charge n'est effectué et que les définitions applicables sont sélectionnées par ordre alphabétique. Pour autoriser un équilibrage de charge, choisissez une valeur comprise entre 1 et 99, où 1 est la pondération la plus faible et 99 la plus élevée. La répartition des messages entre deux ou plusieurs canaux avec des pondérations non nulles est proportionnelle au rapport de ces pondérations. Par exemple, trois canaux avec des valeurs de pondération ClientChannel de 2, 4 et 14 sont sélectionnés à environ 10%, 20% et 70% du temps. Cette distribution n'est pas garantie.

Cet attribut est valide uniquement pour le type de canal de connexion client.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieure à MQCD\_VERSION\_9.

*ClusterPtr (MQPTR)*

Cette zone indique l'adresse d'une liste de noms de cluster.

Si *ClustersDefined* est supérieur à zéro, cette adresse correspond à l'adresse d'une liste de noms de cluster. Le canal appartient à chaque cluster répertorié.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_CLUSSDR ou MQCHT\_CLUSRCVR.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_5.

*ClustersDefined (MQLONG)*

Cette zone indique le nombre de clusters auxquels appartient le canal.

Cette zone indique le nombre de noms de cluster indiqués par *ClusterPtr*. Elle est supérieure ou égale à zéro.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_CLUSSDR ou MQCHT\_CLUSRCVR.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_5.

#### *CLWLChannelPriority (MQLONG)*

Cette zone indique la priorité du canal de charge de travail du cluster.

L'algorithme de sélection du gestionnaire de charge de travail sélectionne une destination dont la priorité est la plus élevée dans l'ensemble de destinations sélectionnées en fonction du rang. S'il existe deux gestionnaires de files d'attente de destination possibles, cet attribut peut être utilisé pour effectuer une reprise en ligne d'un gestionnaire de files d'attente sur l'autre gestionnaire de files d'attente. Tous les messages sont envoyés au gestionnaire de files d'attente avec la priorité la plus élevée jusqu'à la fin, puis les messages sont envoyés au gestionnaire de files d'attente avec la priorité la plus élevée suivante.

La valeur est comprise entre 0 et 9. La valeur par défaut est 0.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_8.

Pour plus d'informations, voir [Configuration d'un cluster de gestionnaires de files d'attente](#).

#### *CLWLChannelRank (MQLONG)*

Cette zone indique le rang du canal de charge de travail du cluster.

L'algorithme de sélection du gestionnaire de charge de travail sélectionne une destination avec le rang le plus élevé. Lorsque la destination finale est un gestionnaire de files d'attente sur un autre cluster, vous pouvez définir le rang des gestionnaires de files d'attente de passerelle intermédiaires (à l'intersection des clusters voisins) de sorte que l'algorithme de sélection choisisse correctement un gestionnaire de files d'attente de destination plus proche de la destination finale.

La valeur est comprise entre 0 et 9. La valeur par défaut est 0.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_8.

Pour plus d'informations, voir [Configuration d'un cluster de gestionnaires de files d'attente](#).

#### *CLWLChannelWeight (MQLONG)*

Cette zone indique le poids du canal de charge de travail du cluster.

Poids du canal de charge de travail du cluster.

L'algorithme de sélection du gestionnaire de charge de travail utilise l'attribut "weight" du canal pour fausser le choix de destination afin que davantage de messages puissent être envoyés à une machine particulière. Par exemple, vous pouvez attribuer à un canal sur un serveur UNIX de grande taille une "pondération" supérieure à celle d'un autre canal sur un PC de bureau de petite taille, et l'algorithme de sélection choisit le serveur UNIX plus souvent que le PC.

La valeur est comprise entre 1 et 99. La valeur par défaut est 50.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_8.

Pour plus d'informations, voir [Configuration d'un cluster de gestionnaires de files d'attente](#).

#### *ConnectionAffinity (MQLONG)*

Cette zone indique si les applications client qui se connectent plusieurs fois à l'aide du même nom de gestionnaire de files d'attente utilisent le même canal client.

Utilisez cet attribut lorsque plusieurs définitions de canal applicables sont disponibles.

La valeur est l'une des suivantes :

## **MQCAFTY\_PREFERENTE\_PRÉFÉRÉE**

La première connexion dans un processus de lecture d'une table de définition de canal du client (CCDT) crée une liste de définitions applicables en fonction de la pondération avec les éventuelles définitions CLNTWGHT (0) applicables en premier et par ordre alphabétique. Chaque connexion du processus tente de se connecter en utilisant la première définition de la liste. Si la connexion échoue, la définition suivante est utilisée. Les définitions ayant échoué avec des valeurs CLNTWGHT autres que 0 sont déplacées vers la fin de la liste. Les définitions CLNTWGHT(0) restent en début de liste et sont sélectionnées en premier pour chaque connexion.

Chaque processus client portant le même nom d'hôte crée toujours la même liste.

Pour les applications client écrites en C, C++ ou l'infrastructure de programmation .NET (y compris .NET entièrement gérée), la liste est mise à jour si la table de définition de canal du client a été modifiée depuis la création de la liste.

Cette valeur est la valeur par défaut.

## **MQCAFTY\_NONE**

La première connexion dans un processus de lecture d'une table de définition de canal du client (CCDT) crée une liste de définitions applicables. Toutes les connexions dans un processus sélectionnent une définition applicable en fonction de la pondération avec toute définition applicable CLNTWGHT(0) sélectionnée en premier et dans l'ordre alphabétique.

Pour les applications client écrites en C, C++ ou l'infrastructure de programmation .NET (y compris .NET entièrement gérée), la liste est mise à jour si la table de définition de canal du client a été modifiée depuis la création de la liste.

Cet attribut est valide uniquement pour le type de canal de connexion client.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieure à MQCD\_VERSION\_9.

### *ConnectionName (MQCHAR264)*

Cette zone indique le nom de connexion du canal.

Pour les canaux récepteurs de cluster (lorsqu'ils sont spécifiés), CONNAME est lié au gestionnaire de files d'attente local, et pour les autres canaux, il est lié au gestionnaire de files d'attente cible. La valeur que vous spécifiez dépend du protocole de transmission (*TransportType*) à utiliser:

- Pour MQXPT\_LU62, il s'agit du nom qualifié complet de l'unité logique partenaire.
- Pour MQXPT\_NETBIOS, il s'agit du nom NetBIOS défini sur la machine distante.
- Pour MQXPT\_TCP, il s'agit soit du nom d'hôte, soit de l'adresse réseau de la machine distante spécifiée au format IPv4 décimal à points ou IPv6 hexadécimal, soit de la machine locale pour les canaux récepteurs de cluster.
- Pour MQXPT\_SPX, il s'agit d'une adresse de style SPX comprenant une adresse réseau de 4 octets, une adresse de noeud de 6 octets et un numéro de socket de 2 octets.

Lors de la définition d'un canal, cette zone n'est pas pertinente pour les canaux dont le *ChannelType* est MQCHT\_SVRCONN ou MQCHT\_RECEIVER. Toutefois, lorsque la définition de canal est transmise à un exit, cette zone contient l'adresse du partenaire, quel que soit le type de canal.

La longueur de cette zone est indiquée par MQ\_CONN\_NAME\_LENGTH. Cette zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_2.

### *DataConversion (MQLONG)*

Cette zone indique si l'agent de canal de transmission de messages tente de convertir les données de message d'application si l'agent de canal de réception de messages ne parvient pas à effectuer cette conversion.

Cette zone s'applique uniquement aux messages qui ne sont pas des segments de messages logiques ; l'agent MCA ne tente jamais de convertir des messages qui sont des segments.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_CLUSSDR ou MQCHT\_CLUSRCVR. Il peut s'agir de :

**MQCDC\_SENDER\_CONVERSION**

Conversion par expéditeur.

**MQCDC\_NO\_SENDER\_CONVERSION**

Pas de conversion par l'expéditeur.

*DefReconnect ( MQLONG)*

L'attribut de canal DefReconnect définit la valeur d'attribut de reconnexion par défaut pour un canal de connexion client.

Option de reconnexion automatique du client par défaut. Vous pouvez configurer un IBM MQ MQI client pour une reconnexion automatique à une application client. Le IBM MQ MQI client essaie de se reconnecter à un gestionnaire de files d'attente après un échec de connexion. Il essaie de se reconnecter sans que l'application client émette un appel MQI MQCONN ou MQCONNX.

La reconnexion est une option MQCONNX . A l'aide de l'attribut de canal DefReconnect , vous pouvez ajouter un comportement de reconnexion à des applications existantes qui utilisent MQCONN. Vous pouvez également modifier le comportement de reconnexion des applications qui utilisent MQCONNX.

Vous pouvez également définir la valeur DefRecon à partir du fichier mqclient .ini pour définir ou modifier le comportement de reconnexion. La valeur DefRecon du fichier mqclient .ini est prioritaire sur l'attribut de canal DefReconnect .

**Syntax**

**DefReconnect** ( MQRCN\_NO (default) |MQRCN\_YES|MQRCN\_Q\_MGR|MQRCN\_DISABLED )

**Paramètres**

**MQRCN\_NO**

MQRCN\_NO est la valeur par défaut.

A moins d'être remplacé par **MQCONNX**, le client n'est pas reconnecté automatiquement.

**MQRCN\_YES**

A moins d'être remplacé par **MQCONNX**, le client se reconnecte automatiquement.

**MQRCN\_Q\_MGR**

Sauf s'il est remplacé par **MQCONNX**, le client se reconnecte automatiquement, mais uniquement au même gestionnaire de files d'attente. L'option QMGR a le même effet que MQCNO\_RECONNECT\_Q\_MGR.

**MQRCN\_DISABLED**

La reconnexion est désactivée, même si elle est demandée par le programme client à l'aide de l'appel **MQCONNX** MQI.

La reconnexion automatique du client n'est pas prise en charge par IBM MQ classes for Java.

*Tableau 822. La reconnexion automatique dépend des valeurs définies dans l'application et dans la définition de canal*

<b>DefReconnect</b>	<b>Options de reconnexion définies dans l'application</b>			
	MQCNO_RECONNE CT	MQCNO_RECONNE CT_Q_MGR	MQCNO_RECONNE CT_AS_DEF	MQCNO_RECONNE CT_DISABLED
MQRCN_NO	YES	QMGR	NO	NO
MQRCN_YES	YES	QMGR	YES	NO
MQRCN_Q_MGR	YES	QMGR	QMGR	NO
MQRCN_DISABLED	NO	NO	NO	NO



## Concepts associés

[reconnexion client automatique](#)

[Reconnexion canal et client](#)

[Strophe CHANNELS du fichier de configuration client](#)

## Référence associée

[«Options \(MQLONG\) pour MQCNO», à la page 331](#)

Options qui contrôlent l'action de MQCONNX.

### Desc (MQCHAR64)

Cette zone peut être utilisée pour les commentaires descriptifs.

Le contenu de la zone n'est pas significatif pour les agents de canal de message. Toutefois, il ne doit contenir que des caractères pouvant être affichés. Elle ne peut pas contenir de caractères nuls ; si nécessaire, elle est remplie à droite avec des blancs. Dans une installation DBCS, la zone peut contenir des caractères DBCS (avec une longueur de zone maximale de 64 octets).

**Remarque :** Si cette zone contient des caractères qui ne font pas partie du jeu de caractères du gestionnaire de files d'attente (tel que défini par l'attribut de gestionnaire de files d'attente **CodedCharSetId**), ces caractères peuvent être convertis de manière incorrecte si cette zone est envoyée à un autre gestionnaire de files d'attente.

La longueur de cette zone est donnée par MQ\_CHANNEL\_DESC\_LENGTH.

### DiscInterval (MQLONG)

Cette zone indique la durée maximale, en secondes, pendant laquelle le canal attend l'arrivée d'un message dans la file d'attente de transmission avant d'arrêter le canal.

En d'autres termes, il spécifie l'intervalle de déconnexion.

La valeur A de zéro entraîne une attente indéfinie de l'agent MCA.

Pour les canaux de connexion serveur utilisant le protocole TCP, l'intervalle représente la valeur de déconnexion d'inactivité du client, en secondes. Si une connexion serveur n'a pas reçu de communication de la part de son client partenaire pendant cette durée, elle met fin à la connexion. L'intervalle d'inactivité de la connexion serveur ne s'applique qu'entre les appels d'API IBM MQ d'un client. Par conséquent, aucun client n'est déconnecté lors d'un appel MQGET à exécution longue avec attente.

Cet attribut n'est pas applicable aux canaux de connexion serveur utilisant des protocoles autres que TCP.

Cette zone est pertinente uniquement pour les canaux avec un *ChannelType* MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_CLUSSDR, MQCHT\_CLUSRCVR ou MQCHT\_SVRCONN.

### ExitDataLongueur (MQLONG)

Cette zone indique la longueur en octets de chaque élément de données utilisateur dans les listes d'éléments de données utilisateur d'exit traités par les zones *MsgUserDataPtr*, *SendUserDataPtr* et *ReceiveUserDataPtr*.

Cette longueur n'est pas nécessairement identique à MQ\_EXIT\_DATA\_LENGTH.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_4.

### ExitNameLongueur (MQLONG)

Cette zone indique la longueur en octets de chacun des noms figurant dans les listes de noms d'exit adressées par les zones *MsgExitPtr*, *SendExitPtr* et *ReceiveExitPtr*.

Cette longueur n'est pas nécessairement identique à MQ\_EXIT\_NAME\_LENGTH.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_4.

### Liste HdrComp[ 2 ] (MQLONG)

Cette zone indique la liste des techniques de compression de données d'en-tête prises en charge par le canal.

La liste contient une ou plusieurs des valeurs suivantes:

**MQCOMPRESS\_NONE**

Aucune compression de données d'en-tête n'est effectuée.

**MQCOMPRESS\_SYSTEM**

La compression de données d'en-tête est effectuée.

**MQCOMPRESS\_NON\_DISPONIBLE**

Les valeurs inutilisées dans la liste sont définies sur cette valeur.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_8.

*HeartbeatInterval (MQLONG)*

Cette zone indique la durée en secondes entre les flux de signaux de présence.

L'interprétation de cette zone dépend du type de canal, comme suit:

- Pour un type de canal MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_RECEIVER MQCHT\_REQUESTER, MQCHT\_CLUSSDR ou MQCHT\_CLUSRCVR, cette zone correspond à la durée, en secondes, entre les flux de pulsations transmis à partir de l'agent MCA émetteur lorsqu'il n'y a pas de message dans la file d'attente de transmission. Cela donne à l'agent MCA récepteur la possibilité de mettre le canal au repos. Pour être utile, *HeartbeatInterval* doit être inférieur à *DiscInterval*.
- Pour un type de canal MQCHT\_CLNTCONN ou MQCHT\_SVRCONN avec la zone Conversations de partage MQCD définie sur zéro, cette zone correspond à la durée, en secondes, entre les flux de pulsations transmis à partir de l'agent MCA serveur lorsque cet agent MCA a émis un appel MQGET avec l'option MQGMO\_WAIT pour le compte d'une application client. Cela permet à l'agent MCA du serveur de gérer les situations où la connexion client échoue lors d'une opération MQGET avec MQGMO\_WAIT.
- Pour un type de canal MQCHT\_CLNTCONN ou MQCHT\_SVRCONN avec la zone Conversations de partage MQCD définie sur une valeur différente de zéro, cette zone correspond à la durée, en secondes, entre deux flux de pulsations lorsqu'aucun flux de données n'est envoyé ou reçu. Cela permet au canal d'être mis au repos efficacement.

La valeur est comprise entre 0 et 999 999. La valeur utilisée est la plus grande des valeurs spécifiées côté émetteur et côté récepteur, sauf si la valeur 0 est spécifiée de part et d'autre, auquel cas aucun échange de signal de présence n'a lieu.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_4.

*KeepAliveIntervalle (MQLONG)*

Cette zone indique la valeur transmise à la pile de communications pour la temporisation du signal de présence pour le canal.

La valeur est applicable pour les protocoles de communication TCP/IP et SPX, bien que toutes les implémentations ne prennent pas en charge ce paramètre.

La valeur est comprise entre 0 et 99 999 ; les unités sont des secondes. La valeur zéro indique que le signal de présence du canal n'est pas activé, bien que le signal de présence puisse encore se produire si le signal de présence TCP/IP (plutôt que le signal de présence du canal) est activé. La valeur spéciale suivante est également admise:

**MQKAI\_AUTO**

Automatique.

Cette valeur indique que l'intervalle de signal de présence est calculé à partir de l'intervalle de signal de présence négocié, comme suit:

- Si l'intervalle des pulsations négocié est supérieur à zéro, l'intervalle des pulsations utilisé est l'intervalle des pulsations plus 60 secondes.
- Si l'intervalle des signaux de présence négocié est égal à zéro, l'intervalle des signaux de présence utilisé est égal à zéro.

- Sous z/OS, le signal de présence TCP/IP se produit lorsque TCPKEEP (YES) est spécifié sur l'objet gestionnaire de files d'attente.
- Dans d'autres environnements, le signal de présence TCP/IP se produit lorsque le paramètre **KEEPALIVE=YES** est spécifié dans la section TCP du fichier de configuration de la mise en file d'attente répartie.

Cette zone est pertinente uniquement pour les canaux dont le *TransportType* est MQXPT\_TCP ou MQXPT\_SPX.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_7.

#### *LocalAddress (MQCHAR48)*

Cette zone indique l'adresse TCP/IP locale définie pour le canal pour les communications sortantes.

Cette zone est vide si aucune adresse spécifique n'est définie pour les communications sortantes. L'adresse peut éventuellement inclure un numéro de port ou une plage de numéros de port. Le format de cette adresse est le suivant:

```
[ip-addr][(low-port[, high-port])]
```

où les crochets ( [ ] ) désignent des informations facultatives, *ip-addr* est spécifié en IPv4 notation décimale à point, IPv6 hexadécimale ou alphanumérique, et *low-port* et *high-port* sont des numéros de port entre parenthèses. Tous sont facultatifs.

Une adresse IP, un port ou une plage de ports spécifique pour les communications sortantes est utile dans les scénarios de récupération où un canal est redémarré sur une pile TCP/IP différente.

La forme de *LocalAddress* est similaire à *ConnectionName*, mais ne doit pas être confondue avec celle-ci. *LocalAddress* indique les caractéristiques des communications locales, tandis que *ConnectionName* indique comment atteindre un gestionnaire de files d'attente éloignées.

Depuis la IBM MQ 9.3.0, l'interface JMQUI ( Java Message Queueing Interface) a été mise à jour pour s'assurer que la zone d'adresse locale est définie sur un objet MQCD après la création d'une instance de canal et qu'elle est connectée à un gestionnaire de files d'attente. Cela signifie que lorsqu'une sortie de canal écrite en Java appelle la méthode MQCD.*getLocalAddress()*, la méthode renvoie l'adresse locale que l'instance de canal utilise. Avant la IBM MQ 9.3.0, l'exit de sécurité de canal n'était pas parvenu à accéder à l'adresse locale utilisée par l'instance de canal et la méthode MQCD.*getLocalAddress()* a renvoyé la valeur null.

Cette zone est pertinente uniquement pour les canaux avec un *TransportType* de MQXPT\_TCP et un *ChannelType* de MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_REQUESTER, MQCHT\_CLNTCONN, MQCHT\_CLUSSDR ou MQCHT\_CLUSRCVR.

La longueur de cette zone est indiquée par MQ\_LOCAL\_ADDRESS\_LENGTH. Cette zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_7.

#### *LongMCAUserIdLength (MQLONG)*

Cette zone indique la longueur en octets de l'identificateur utilisateur MCA complet désigné par *LongMCAUserIdPtr*.

Cette zone n'est pas pertinente pour les canaux avec un *ChannelType* de MQCHT\_CLNTCONN.

Il s'agit d'une zone d'entrée-sortie pour l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_6.

#### *LongMCAUserIdPtr (MQPTR)*

Cette zone indique l'adresse de l'ID utilisateur MCA long.

Si *LongMCAUserIdLength* est supérieur à zéro, cette zone correspond à l'adresse de l'ID utilisateur MCA complet. La longueur de l'identificateur complet est indiquée par *LongMCAUserIdLength*. Les 12 premiers octets de l'identificateur utilisateur MCA sont également contenus dans la zone *MCAUserIdentifier*.

Pour plus de détails sur l'ID utilisateur MCA, reportez-vous à la description de la zone *MCAUserIdentifier*.

Cette zone n'est pas pertinente pour les canaux avec un *ChannelType* de MQCHT\_SDR, MQCHT\_SVR, MQCHT\_CLNTCONN ou MQCHT\_CLUSSDR.

Il s'agit d'une zone d'entrée-sortie pour l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_6.

#### *LongRemoteUserIdLongueur (MQLONG)*

Cette zone indique la longueur en octets de l'ID utilisateur distant complet désigné par *LongRemoteUserIdPtr*.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_CLNTCONN ou MQCHT\_SVRCONN.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_6.

#### *LongRemoteUserIdPtr (MQPTR)*

Cette zone indique l'adresse de l'ID utilisateur éloigné long.

Si *LongRemoteUserIdLength* est supérieur à zéro, cet indicateur correspond à l'adresse de l'ID utilisateur distant complet. La longueur de l'identificateur complet est indiquée par *LongRemoteUserIdLength*. Les 12 premiers octets de l'ID utilisateur distant sont également contenus dans la zone *RemoteUserIdentifier*.

Pour plus de détails sur l'ID utilisateur distant, voir la description de la zone *RemoteUserIdentifier*.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_CLNTCONN ou MQCHT\_SVRCONN.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_6.

#### *LongRetry(MQLONG)*

Cette zone indique le nombre utilisé après épuisement du nombre indiqué par *ShortRetryCount*.

Il indique le nombre maximal de nouvelles tentatives de connexion à la machine distante, à des intervalles définis par *LongRetryInterval*, avant la consignation d'une erreur à l'opérateur.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_CLUSSDR ou MQCHT\_CLUSRCVR.

#### *Intervalle LongRetry(MQLONG)*

Cette zone indique le nombre maximal de secondes à attendre avant de tenter à nouveau la connexion à la machine distante.

L'intervalle entre les nouvelles tentatives peut être étendu si le canal doit attendre pour devenir actif.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_CLUSSDR ou MQCHT\_CLUSRCVR.

#### *MaxInstances (MQLONG)*

Cette zone indique le nombre maximal d'instances simultanées d'un canal de connexion serveur individuel pouvant être démarrées.

Cette zone est utilisée uniquement sur les canaux de connexion serveur.

La valeur de la zone peut être comprise entre 0 et 999 999 999. La valeur 0 (zéro) empêche tout accès client.

La valeur par défaut de cette zone est 999 999 999.

Si la valeur de cette zone est réduite à un nombre inférieur au nombre d'instances du canal de connexion serveur en cours d'exécution, les instances en cours d'exécution ne sont pas affectées. Toutefois, les nouvelles instances ne peuvent pas démarrer tant qu'un nombre suffisant d'instances existantes n'ont pas cessé de s'exécuter, de sorte que le nombre d'instances en cours d'exécution est inférieur à la valeur de la zone.

*MaxInstancesPerClient (MQLONG)*

Cette zone indique le nombre maximal d'instances simultanées d'un canal de connexion serveur individuel pouvant être démarrées à partir d'un seul client.

Dans ce contexte, les connexions émanant de la même adresse réseau distante peuvent être considérées comme étant issues du même client.

Cette zone est utilisée uniquement sur les canaux de connexion serveur.

La valeur de la zone peut être comprise entre 0 et 999 999 999. La valeur 0 (zéro) empêche tout accès client.

La valeur par défaut de cette zone est 999 999 999.

Si la valeur de cette zone est réduite à un nombre inférieur au nombre d'instances du canal de connexion serveur en cours d'exécution à partir de clients individuels, les instances en cours d'exécution ne sont pas affectées. Toutefois, les nouvelles instances de l'un de ces clients ne peuvent pas démarrer tant qu'un nombre suffisant d'instances existantes n'ont pas cessé de s'exécuter, de sorte que le nombre d'instances en cours d'exécution, provenant du client qui tente d'en démarrer une nouvelle, est inférieur à la valeur de la zone.

*Longueur MaxMsg(MQLONG)*

Cette zone indique la longueur maximale des messages pouvant être transmis sur le canal.

Cette valeur est comparée à celle du canal éloigné et la valeur la plus faible des deux est le maximum réel.

*MCAName (MQCHAR20)*

Cette zone est une zone réservée.

La valeur de cette zone est vide.

La longueur de cette zone est indiquée par MQ\_MCA\_NAME\_LENGTH.

*MCASecurityId (MQBYTE40)*

Cette zone indique l'identificateur de sécurité de l'agent MCA.

Cette zone n'est pas pertinente pour les canaux avec un *ChannelType* de MQCHT\_CLNTCONN.

La valeur spéciale suivante indique qu'il n'y a pas d'identificateur de sécurité:

**MQSID\_AUCUN**

Aucun identificateur de sécurité n'a été indiqué.

La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQSID\_NONE\_ARRAY est également définie ; cette constante a la même valeur que MQSID\_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit d'une zone d'entrée-sortie pour l'exit. La longueur de cette zone est indiquée par MQ\_SECURITY\_ID\_LENGTH. Cette zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_6.

*MCAType (MQLONG)*

Cette zone indique le type de programme d'agent MCA.

Cette zone s'applique uniquement aux canaux dont le *ChannelType* est MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_REQUESTER, MQCHT\_CLUSSDR ou MQCHT\_CLUSRCVR.

La valeur est l'une des suivantes :

### **PROCESSUS MQMCAT\_PROCESS**

L'agent MCA fonctionne en tant que processus distinct.

### **MQMCAT\_THREAD**

Unité d'exécution (Multiplateformes).

L'agent MCA fonctionne en tant qu'unité d'exécution distincte.

Cette zone n'est pas présente lorsque *Version* est inférieure à MQCD\_VERSION\_2.

### *MCAUserIdentifier (MQCHAR12)*

Cette zone indique l'ID utilisateur de l'agent MCA.

Cette zone utilise les 12 premiers octets de l'identificateur utilisateur MCA et peut être définie par un agent de sécurité.

Deux zones contiennent l'identificateur utilisateur MCA:

- *MCAUserIdentifier* contient les 12 premiers octets de l'identificateur utilisateur MCA et est rempli avec des blancs si l'identificateur est inférieur à 12 octets. *MCAUserIdentifier* peut être vide.
- *LongMCAUserIdPtr* pointe vers l'ID utilisateur MCA complet, qui peut dépasser 12 octets. Sa longueur est donnée par *LongMCAUserIdLength*. L'identificateur complet ne contient pas de blancs de fin et n'est pas terminé par une valeur NULL. Si l'identificateur est vide, *LongMCAUserIdLength* est égal à zéro et la valeur de *LongMCAUserIdPtr* n'est pas définie.

**Remarque :** *LongMCAUserIdPtr* n'est pas présent si *Version* est inférieur à MQCD\_VERSION\_6.

Si l'ID utilisateur MCA n'est pas à blanc, il indique l'ID utilisateur à utiliser par l'agent MCA pour l'autorisation d'accès aux ressources IBM MQ. Pour les types de canal MQCHT\_REQUESTER, MQCHT\_RECEIVER et MQCHT\_CLUSRCVR, si PutAuthority est MQPA\_DEFAULT, il s'agit de l'identificateur utilisateur utilisé pour les vérifications d'autorisation de l'opération d'insertion dans les files d'attente de destination.

Si l'ID utilisateur MCA est vide, l'agent MCA utilise son ID utilisateur par défaut.

L'ID utilisateur MCA peut être défini par un exit de sécurité pour indiquer l'ID utilisateur que l'agent MCA doit utiliser. L'exit peut modifier *MCAUserIdentifier* ou la chaîne pointée par *LongMCAUserIdPtr*. Si les deux sont modifiés mais différent l'un de l'autre, l'agent MCA utilise *LongMCAUserIdPtr* de préférence à *MCAUserIdentifier*. Si l'exit modifie la longueur de la chaîne adressée par *LongMCAUserIdPtr*, *LongMCAUserIdLength* doit être défini en conséquence. Si l'exit augmente la longueur de l'identificateur, il doit allouer de la mémoire de la longueur requise, définir cette mémoire sur l'identificateur requis et placer l'adresse de cette mémoire dans *LongMCAUserIdPtr*. L'exit est responsable de la libération de cette mémoire lorsque l'exit est appelé ultérieurement avec la raison MQXR\_TERM.

Pour les canaux avec un *ChannelType* de MQCHT\_SVRCONN, si *MCAUserIdentifier* dans la définition de canal est vide, tout identificateur utilisateur transféré à partir du client est copié dans le canal. Cet identificateur utilisateur (après toute modification apportée par l'exit de sécurité sur le serveur) est celui sous lequel l'application client est supposée s'exécuter.

L'identificateur utilisateur MCA n'est pas pertinent pour les canaux avec un *ChannelType* de MQCHT\_SDR, MQCHT\_SVR, MQCHT\_CLNTCONN, MQCHT\_CLUSSDR.

Il s'agit d'une zone d'entrée-sortie pour l'exit. La longueur de cette zone est indiquée par MQ\_USER\_ID\_LENGTH. Cette zone n'est pas présente lorsque *Version* est inférieur à MQCD\_VERSION\_2.

### *ModeName (MQCHAR8)*

Cette zone indique le nom de mode LU 6.2.

Cette zone est pertinente uniquement si le protocole de transmission (*TransportType*) est MQXPT\_LU62et que *ChannelType* n'est pas MQCHT\_SVRCONN ou MQCHT\_RECEIVER.

Cette zone est toujours vide. Les informations sont contenues dans l'objet côté communications à la place.

La longueur de cette zone est indiquée par `MQ_MODE_NAME_LENGTH`.

*MsgCompListe [ 16 ] (MQLONG)*

Cette zone indique la liste des techniques de compression de données de message prises en charge par le canal.

La liste contient une ou plusieurs des valeurs suivantes:

**MQCOMPRESS\_NONE**

Aucune compression de données de message n'est effectuée.

**MQCOMPRESS\_RLE**

La compression de données de message est effectuée à l'aide de l'algorithme RLE.

**MQCOMPRESS\_ZLIBFAST**

La compression de données de message est effectuée à l'aide de la technique de compression zlib. Il est préférable d'utiliser une durée de compression rapide.

**MQCOMPRESS\_ZLIBHIGH**

La compression de données de message est effectuée à l'aide de la technique de compression zlib. Il est préférable d'utiliser une compression de haut niveau.

**MQCOMPRESS\_ANY**

Toute technique de compression prise en charge par le gestionnaire de files d'attente peut être utilisée pour la compression des messages. `MQCOMPRESS_ANY` n'est valide que pour les canaux de réception, de demandeur et de connexion serveur.

**MQCOMPRESS\_NON\_DISPONIBLE**

Les valeurs inutilisées dans la liste sont définies sur cette valeur.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à `MQCD_VERSION_8`.

*MsgExit (MQCHARn)*

Cette zone indique le nom de l'exit de message de canal.

Si ce nom n'est pas vide, l'exit est appelé aux moments suivants:

- Immédiatement après l'extraction d'un message de la file d'attente de transmission (émetteur ou serveur) ou immédiatement avant l'insertion d'un message dans une file d'attente de destination (récepteur ou demandeur).

L'exit est associé à l'intégralité du message d'application et à l'en-tête de la file d'attente de transmission pour modification.

- Lors de l'initialisation et de l'arrêt du canal.

Cette zone n'est pas pertinente pour les canaux dont le *ChannelType* est `MQCHT_SVRCONN` ou `MQCHT_CLNTCONN`; un exit de message n'est jamais appelé pour ces canaux.

Voir «MQCD-Définition de canal», à la page 1553 pour une description du contenu de cette zone dans divers environnements.

La longueur de cette zone est indiquée par `MQ_EXIT_NAME_LENGTH`.

**Remarque :** La valeur de cette constante est spécifique à l'environnement.

*MsgExitPtr (MQPTR)*

Cette zone indique l'adresse de la première zone *MsgExit*.

Si *MsgExitsDefined* est supérieur à zéro, cette adresse est l'adresse de la liste des noms de chaque exit de message de canal dans la chaîne.

Chaque nom se trouve dans une zone de longueur *ExitNameLength*, complétée à droite par des blancs. Il y a des champs *MsgExitsDefined* adjacents les uns aux autres-un pour chaque sortie.

Toutes les modifications apportées à ces noms par un exit sont conservées, bien que l'exit de canal de transmission de messages n'effectue aucune action explicite-il ne modifie pas les exits qui sont appelés.

Si *MsgExitsDefined* est égal à zéro, cette zone est le pointeur null.

Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_4.

#### *MsgExitsDéfini (MQLONG)*

Cette zone indique le nombre d'exits de message de canal définis dans la chaîne.

Elle est supérieure ou égale à zéro.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_4.

#### *MsgRetry(MQLONG)*

Cette zone indique le nombre de fois où MCA tente d'insérer le message, après l'échec de la première tentative.

Cette zone indique le nombre de fois où l'agent MCA tente l'opération d'ouverture ou d'insertion, si le premier MQOPEN ou MQPUT échoue avec le code achèvement MQCC\_FAILED. L'effet de cet attribut varie selon que *MsgRetryExit* est vide ou non:

- Si *MsgRetryExit* est vide, l'attribut **MsgRetryCount** contrôle si l'agent MCA tente de nouveau d'effectuer des tentatives. Si la valeur de l'attribut est zéro, aucune nouvelle tentative n'est effectuée. Si la valeur de l'attribut est supérieure à zéro, les nouvelles tentatives sont effectuées à des intervalles définis par l'attribut **MsgRetryInterval**.

Les nouvelles tentatives sont effectuées uniquement pour les codes anomalie suivants:

- MQRC\_PAGESET\_FULL
- MQRC\_PUT\_INHIBÉ
- MQRC\_Q\_FULL

Pour les autres codes raison, l'agent MCA passe immédiatement à son traitement d'échec normal, sans relancer le message d'échec.

- Si *MsgRetryExit* n'est pas vide, l'attribut **MsgRetryCount** n'affecte pas l'agent MCA ; il s'agit plutôt de l'exit de relance de message qui détermine le nombre de tentatives de relance et à quels intervalles ; l'exit est appelé même si l'attribut **MsgRetryCount** est égal à zéro.

L'attribut **MsgRetryCount** est mis à la disposition de l'exit dans la structure MQCD, mais l'exit qu'il n'est pas nécessaire de prendre en compte-les nouvelles tentatives se poursuivent indéfiniment jusqu'à ce que l'exit renvoie MQXCC\_SUPPRESS\_FUNCTION dans la zone *ExitResponse* de MQCXP.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_REQUESTER, MQCHT\_RECEIVER ou MQCHT\_CLUSRCVR.

Cette zone n'est pas présente lorsque *Version* est inférieur à MQCD\_VERSION\_3.

#### *Exit MsgRetry(MQCHARn)*

Cette zone indique le nom de l'exit de relance de message de canal.

L'exit de relance de message est un exit appelé par l'agent MCA lorsque ce dernier reçoit le code achèvement MQCC\_FAILED d'un appel MQOPEN ou MQPUT. L'objectif de l'exit est de spécifier un intervalle de temps pendant lequel l'agent MCA attend avant de relancer l'opération MQOPEN ou MQPUT. L'exit peut également être défini pour ne pas relancer l'opération.

L'exit est appelé pour tous les codes raison dont le code achèvement est MQCC\_FAILED-les paramètres de l'exit déterminent les codes raison pour lesquels l'agent MCA doit effectuer une nouvelle tentative, le nombre de tentatives et les intervalles de temps.



Lorsque l'opération ne doit plus être tentée, l'agent MCA exécute son traitement d'échec normal ; ce traitement comprend la génération d'un message de rapport d'exception (s'il est spécifié par l'expéditeur) et le placement du message d'origine dans la file d'attente des messages non livrés ou la suppression du message (selon que l'expéditeur a spécifié MQRO\_DEAD\_LETTER\_Q ou MQRO\_DISCARD\_MSG). Les échecs impliquant la file d'attente de rebut (par exemple, la file d'attente de rebut saturée) n'entraînent pas l'appel de l'exit de relance de message.

Si le nom de l'exit n'est pas vide, l'exit est appelé aux moments suivants:

- Immédiatement avant d'effectuer l'attente avant de tenter de distribuer à nouveau un message
- Lors de l'initialisation et de l'arrêt du canal

Voir «MQCD-Définition de canal», à la page 1553 pour une description du contenu de cette zone dans divers environnements.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_REQUESTER, MQCHT\_RECEIVER ou MQCHT\_CLUSRCVR.

La longueur de cette zone est indiquée par MQ\_EXIT\_NAME\_LENGTH.

**Remarque :** La valeur de cette constante est spécifique à l'environnement.

Cette zone n'est pas présente lorsque *Version* est inférieur à MQCD\_VERSION\_3.

#### *Intervalle MsgRetry(MQLONG)*

Cette zone indique l'intervalle minimal en millisecondes après lequel l'opération d'ouverture ou d'insertion est relancée.

L'effet de cet attribut varie selon que *MsgRetryExit* est vide ou non:

- Si *MsgRetryExit* est vide, l'attribut **MsgRetryInterval** indique la période minimale pendant laquelle l'agent MCA attend avant de relancer un message, si le premier MQOPEN ou MQPUT échoue avec le code achèvement MQCC\_FAILED. La valeur zéro signifie que la nouvelle tentative sera effectuée dès que possible après la tentative précédente. Les nouvelles tentatives sont effectuées uniquement si *MsgRetryCount* est supérieur à zéro.

Cet attribut est également utilisé comme temps d'attente si l'exit de relance de message renvoie une valeur non valide dans la zone *MsgRetryInterval* de MQCXP.

- Si *MsgRetryExit* n'est pas vide, l'attribut **MsgRetryInterval** n'affecte pas l'agent MCA ; c'est plutôt l'exit de relance de message qui détermine la durée d'attente de l'agent MCA. L'attribut **MsgRetryInterval** est mis à la disposition de l'exit dans la structure MQCD, mais l'exit n'est pas requis pour l'honorer.

La valeur est comprise entre 0 et 999 999 999.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_REQUESTER, MQCHT\_RECEIVER ou MQCHT\_CLUSRCVR.

Cette zone n'est pas présente lorsque *Version* est inférieur à MQCD\_VERSION\_3.

Les zones suivantes de cette structure ne sont pas présentes si *Version* est inférieur à MQCD\_VERSION\_4.

#### *MsgRetryUserData (MQCHAR32)*

Cette zone indique les données utilisateur de l'exit de relance de message de canal.

Ces données sont transmises à l'exit de relance de message de canal dans la zone *ExitData* du paramètre **ChannelExitParms** (voir MQ\_CHANNEL\_EXIT).

Cette zone contient initialement les données qui ont été définies dans la définition de canal. Toutefois, pendant la durée de vie de cette instance MCA, toutes les modifications apportées au contenu de cette zone par un exit de n'importe quel type sont conservées par l'agent MCA et rendues visibles par les appels ultérieurs des exits (quel que soit le type) pour cette instance MCA. Ces modifications n'affectent pas la définition de canal utilisée par d'autres instances MCA. Tous les caractères (y compris les données binaires) peuvent être utilisés.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_REQUESTER, MQCHT\_RECEIVER ou MQCHT\_CLUSRCVR.

La longueur de cette zone est donnée par MQ\_EXIT\_DATA\_LENGTH. Cette zone n'est pas présente lorsque *Version* est inférieur à MQCD\_VERSION\_3.

Cette zone n'est pas pertinente dans IBM MQ for IBM i.

#### *Données MsgUser(MQCHAR32)*

Cette zone indique les données utilisateur de l'exit de message de canal.

Ces données sont transmises à l'exit de message de canal dans la zone *ExitData* du paramètre **ChannelExitParms** (voir MQ\_CHANNEL\_EXIT).

Cette zone contient initialement les données qui ont été définies dans la définition de canal. Toutefois, pendant la durée de vie de cette instance MCA, toutes les modifications apportées au contenu de cette zone par un exit de n'importe quel type sont conservées par l'agent MCA et rendues visibles par les appels ultérieurs des exits (quel que soit le type) pour cette instance MCA. Ces modifications n'affectent pas la définition de canal utilisée par d'autres instances MCA. Tous les caractères (y compris les données binaires) peuvent être utilisés.

La longueur de cette zone est donnée par MQ\_EXIT\_DATA\_LENGTH.

Cette zone n'est pas pertinente dans IBM MQ for IBM i.

#### *MsgUserDataPtr (MQPTR)*

Cette zone indique l'adresse de la première zone *MsgUserData*.

Si *MsgExitsDefined* est supérieur à zéro, cette adresse correspond à l'adresse de la liste des éléments de données utilisateur pour chaque exit de message de canal de la chaîne.

Chaque élément de données utilisateur se trouve dans une zone de longueur *ExitDataLength*, complétée à droite par des blancs. Il y a des champs *MsgExitsDefined* adjacents les uns aux autres-un pour chaque sortie. Si le nombre d'éléments de données utilisateur définis est inférieur au nombre de noms d'exit, les éléments de données utilisateur non définis sont mis à blanc. A l'inverse, si le nombre d'éléments de données utilisateur définis est supérieur au nombre de noms d'exit, les éléments de données utilisateur excédentaires sont ignorés et ne sont pas présentés à l'exit.

Toutes les modifications apportées à ces valeurs par un exit sont conservées. Cela permet à un exit de transmettre des informations à un autre exit. Aucune validation n'est effectuée sur les modifications. Par exemple, des données binaires peuvent être écrites dans ces zones si nécessaire.

Si *MsgExitsDefined* est égal à zéro, cette zone est le pointeur null.

Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_4.

#### *NetworkPriority (MQLONG)*

Cette zone indique la priorité de la connexion réseau pour le canal.

Lorsque plusieurs chemins d'accès à une destination particulière sont disponibles, le chemin ayant la priorité la plus élevée est choisi. La valeur est comprise entre 0 et 9 ; 0 est la priorité la plus basse.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_CLUSSDR ou MQCHT\_CLUSRCVR.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_5.

Les zones suivantes de cette structure ne sont pas présentes si *Version* est inférieur à MQCD\_VERSION\_6.

### *NonPersistentMsgSpeed (MQLONG)*

Cette zone indique la vitesse à laquelle les messages non persistants transitent par le canal.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_RECEIVER, MQCHT\_REQUESTER, MQCHT\_CLUSSDR ou MQCHT\_CLUSRCVR.

La valeur est l'une des suivantes :

#### **MQNPMS\_NORMAL**

Vitesse normale.

Si un canal est défini pour être MQNPMS\_NORMAL, les messages non persistants transitent par le canal à une vitesse normale. Cela présente l'avantage que ces messages ne sont pas perdus en cas de défaillance du canal. En outre, les messages persistants et non persistants de la même file d'attente de transmission conservent leur ordre relatif les uns par rapport aux autres.

#### **MQNPMS\_FAST**

Vitesse rapide.

Si un canal est défini comme MQNPMS\_FAST, les messages non persistants transitent par le canal à vitesse rapide. Cela améliore le débit du canal, mais signifie que les messages non persistants sont perdus en cas de défaillance du canal. En outre, il est possible que les messages non persistants passent avant les messages persistants en attente dans la même file d'attente de transmission, c'est-à-dire que l'ordre des messages non persistants n'est pas conservé par rapport aux messages persistants. Toutefois, l'ordre des messages non persistants les uns par rapport aux autres est conservé. De même, l'ordre des messages persistants les uns par rapport aux autres est conservé.

### *Mot de passe (MQCHAR12)*

Cette zone indique le mot de passe utilisé par l'agent MCA lors de la tentative de lancement d'une session SNA sécurisée avec un agent MCA éloigné.

Cette zone ne peut être vide que sous AIX, Linux, and Windowset n'est pertinente que pour les canaux dont le *ChannelType* est MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_REQUESTER ou MQCHT\_CLNTCONN. Sous z/OS, cette zone n'est pas pertinente.

La longueur de cette zone est indiquée par MQ\_PASSWORD\_LENGTH. Cependant, seuls les 10 premiers caractères sont utilisés.

Cette zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_2.

### *PropertyControl (MQLONG)*

Cette zone indique ce qu'il advient des propriétés des messages lorsque le message est sur le point d'être envoyé à un gestionnaire de files d'attente V6 ou antérieure (gestionnaire de files d'attente qui ne comprend pas le concept de descripteur de propriété).

La valeur peut être l'une des valeurs suivantes :

#### **COMPATIBILITE\_MQPROP\_COMPATIBILITÉ**

Si le message contient une propriété avec le préfixe **mcd.**, **jms.**, **usr.** ou **mqext.**, toutes les propriétés de message sont distribuées à l'application dans un en-tête MQRFH2. Sinon, toutes les propriétés du message, à l'exception de celles contenues dans le descripteur de message (ou l'extension), sont supprimées et ne sont plus accessibles à l'application.

Cette valeur est la valeur par défaut ; elle permet aux applications, qui s'attendent à ce que les propriétés liées à JMSse trouvent dans un en-tête MQRFH2 dans les données de message, de continuer à fonctionner sans modification.

#### **MQPROP\_NONE**

Toutes les propriétés du message, à l'exception de celles du descripteur de message (ou extension), sont supprimées du message avant son envoi au gestionnaire de files d'attente éloignées.

## **MQPROP\_ALL**

Toutes les propriétés du message sont incluses dans le message lorsqu'il est envoyé au gestionnaire de files d'attente éloignées. Les propriétés, hormis celles associées au descripteur de message (ou à l'extension), sont placées dans un ou plusieurs en-têtes MQRFH2 dans les données du message.

Cet attribut est applicable aux canaux émetteur, serveur, émetteur de cluster et récepteur de cluster.

«MQIA\_\* (Sélecteurs d'attribut d'entier)», à la page 130

«MQPROP\_\* (Valeurs de contrôle de propriété de file d'attente et de canal et longueur maximale des propriétés)», à la page 171

### *PutAuthority (MQLONG)*

Cette zone indique si l'ID utilisateur dans les informations contextuelles associées à un message est utilisé pour établir le droit d'insertion du message dans la file d'attente de destination.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_REQUESTER, MQCHT\_RECEIVER ou MQCHT\_CLUSRCVR. Il peut s'agir de :

### **MQPA\_PAR DEFAUT**

L'ID utilisateur par défaut est utilisé.

### **MQPA\_CONTEXT**

L'ID utilisateur de contexte est utilisé.

### **MQPA\_ALTERNATE\_OR\_MCA**

L'ID utilisateur de la zone UserIdentifier du descripteur de message est utilisé. Aucun ID utilisateur reçu du réseau n'est utilisé. Cette valeur est prise en charge uniquement sous z/OS.

### **MQPA\_ONLY\_MCA**

L'ID utilisateur par défaut est utilisé. Aucun ID utilisateur reçu du réseau n'est utilisé. Cette valeur est prise en charge uniquement sous z/OS.

### *QMgrName (MQCHAR48)*

Cette zone indique le nom du gestionnaire de files d'attente auquel un exit peut se connecter.

Pour les canaux avec une *ChannelType* autre que MQCHT\_CLNTCONN, cette zone correspond au nom du gestionnaire de files d'attente auquel un exit peut se connecter, qui sous AIX, Linux, and Windows, est toujours non vide.

La longueur de cette zone est indiquée par MQ\_Q\_MGR\_NAME\_LENGTH.

### *ReceiveExit (MQCHARn)*

Cette zone indique le nom de l'exit de réception de canal.

Si ce nom n'est pas vide, l'exit est appelé aux moments suivants:

- Immédiatement avant le traitement des données réseau reçues.

L'exit reçoit la mémoire tampon de transmission complète telle qu'elle a été reçue. Le contenu de la mémoire tampon peut être modifié selon les besoins.

- Lors de l'initialisation et de l'arrêt du canal.

Voir «MQCD-Définition de canal», à la page 1553 pour une description du contenu de cette zone dans divers environnements.

La longueur de cette zone est indiquée par MQ\_EXIT\_NAME\_LENGTH.

**Remarque :** La valeur de cette constante est spécifique à l'environnement.

### *ReceiveExitPtr (MQPTR)*

Cette zone indique l'adresse de la première zone *ReceiveExit*.

Si *ReceiveExitsDefined* est supérieur à zéro, cette adresse est l'adresse de la liste des noms de chaque exit de réception de canal dans la chaîne.

Chaque nom se trouve dans une zone de longueur *ExitNameLength*, complétée à droite par des blancs. Il y a des champs *ReceiveExitsDefined* adjacents les uns aux autres-un pour chaque sortie.

Toutes les modifications apportées à ces noms par un exit sont conservées, bien que l'exit de canal de transmission de messages n'effectue aucune action explicite-il ne modifie pas les exits qui sont appelés.

Si *ReceiveExitsDefined* est égal à zéro, cette zone est le pointeur null.

Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_4.

#### *ReceiveExitsDéfini (MQLONG)*

Cette zone indique le nombre d'exits de réception de canal définis dans la chaîne.

Elle est supérieure ou égale à zéro.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_4.

#### *Données ReceiveUser(MQCHAR32)*

Ce canal spécifie les données utilisateur d'exit de réception de canal.

Ces données sont transmises à l'exit de réception de canal dans la zone *ExitData* du paramètre **ChannelExitParms** (voir MQ\_CHANNEL\_EXIT).

Cette zone contient initialement les données qui ont été définies dans la définition de canal. Toutefois, pendant la durée de vie de cette instance MCA, toutes les modifications apportées au contenu de cette zone par un exit de n'importe quel type sont conservées par l'agent MCA et rendues visibles par les appels ultérieurs des exits (quel que soit le type) pour cette instance MCA. Cela s'applique aux exits sur différentes conversations. Ces modifications n'affectent pas la définition de canal utilisée par d'autres instances MCA. Tous les caractères (y compris les données binaires) peuvent être utilisés.

La longueur de cette zone est donnée par MQ\_EXIT\_DATA\_LENGTH.

Cette zone n'est pas pertinente dans IBM MQ for IBM i.

Les zones suivantes de cette structure ne sont pas présentes si *Version* est inférieur à MQCD\_VERSION\_2.

#### *ReceiveUserDataPtr (MQPTR)*

Cette zone indique l'adresse de la première zone *ReceiveUserData*.

Si *ReceiveExitsDefined* est supérieur à zéro, cette adresse correspond à l'adresse de la liste des éléments de données utilisateur pour chaque exit de réception de canal de la chaîne.

Chaque élément de données utilisateur se trouve dans une zone de longueur *ExitDataLength*, complétée à droite par des blancs. Il y a des champs *ReceiveExitsDefined* adjacents les uns aux autres-un pour chaque sortie. Si le nombre d'éléments de données utilisateur définis est inférieur au nombre de noms d'exit, les éléments de données utilisateur non définis sont mis à blanc. A l'inverse, si le nombre d'éléments de données utilisateur définis est supérieur au nombre de noms d'exit, les éléments de données utilisateur excédentaires sont ignorés et ne sont pas présentés à l'exit.

Toutes les modifications apportées à ces valeurs par un exit sont conservées. Cela permet à un exit de transmettre des informations à un autre exit. Aucune validation n'est effectuée sur les modifications. Par exemple, des données binaires peuvent être écrites dans ces zones si nécessaire.

Si *ReceiveExitsDefined* est égal à zéro, cette zone est le pointeur null.

Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_4.

Les zones suivantes de cette structure ne sont pas présentes si *Version* est inférieur à MQCD\_VERSION\_5.

#### *RemotePassword (MQCHAR12)*

Cette zone indique le mot de passe d'un partenaire.

Cette zone contient des informations valides uniquement si *ChannelType* est MQCHT\_CLNTCONN ou MQCHT\_SVRCONN.

- Pour un exit de sécurité sur un canal MQCHT\_CLNTCONN, ce mot de passe est un mot de passe obtenu de l'environnement. L'exit peut choisir de l'envoyer à l'exit de sécurité sur le serveur.
- Pour un exit de sécurité sur un canal MQCHT\_SVRCONN, cette zone peut contenir un mot de passe obtenu à partir de l'environnement sur le client, s'il n'y a pas d'exit de sécurité client. L'exit peut utiliser ce mot de passe pour valider l'identificateur utilisateur dans *RemoteUserIdentifier*.

S'il existe un exit de sécurité sur le client, ces informations peuvent être obtenues dans un flux de sécurité à partir du client.

La longueur de cette zone est indiquée par MQ\_PASSWORD\_LENGTH. Cette zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_2.

#### *ID RemoteSecurity(MQBYTE40)*

Cette zone indique l'identificateur de sécurité de l'utilisateur distant.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_CLNTCONN ou MQCHT\_SVRCONN.

La valeur spéciale suivante indique qu'il n'y a pas d'identificateur de sécurité:

#### **MQSID\_AUCUN**

Aucun identificateur de sécurité n'a été indiqué.

La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQSID\_NONE\_ARRAY est également définie ; cette constante a la même valeur que MQSID\_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit d'une zone d'entrée de l'exit. La longueur de cette zone est indiquée par MQ\_SECURITY\_ID\_LENGTH. Cette zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_6.

Les zones suivantes de cette structure ne sont pas présentes si *Version* est inférieur à MQCD\_VERSION\_7.

#### *Identificateur RemoteUser(MQCHAR12)*

Cette zone indique les 12 premiers octets d'un ID utilisateur d'un partenaire.

Deux zones contiennent l'identificateur d'utilisateur distant:

- *RemoteUserIdentifier* contient les 12 premiers octets de l'identificateur d'utilisateur distant et est rempli avec des blancs si l'identificateur est inférieur à 12 octets. *RemoteUserIdentifier* peut être vide.
- *LongRemoteUserIdPtr* pointe vers l'ID utilisateur distant complet, qui peut dépasser 12 octets. Sa longueur est donnée par *LongRemoteUserIdLength*. L'identificateur complet ne contient pas de blancs de fin et n'est pas terminé par une valeur NULL. Si l'identificateur est vide, *LongRemoteUserIdLength* est égal à zéro et la valeur de *LongRemoteUserIdPtr* n'est pas définie. *LongRemoteUserIdPtr* n'est pas présent si *Version* est inférieur à MQCD\_VERSION\_6.

L'identificateur d'utilisateur distant est pertinent uniquement pour les canaux dont le *ChannelType* est MQCHT\_CLNTCONN ou MQCHT\_SVRCONN.

- Pour un exit de sécurité sur un canal MQCHT\_CLNTCONN, cette valeur est un identificateur utilisateur obtenu de l'environnement. L'exit peut choisir de l'envoyer à l'exit de sécurité sur le serveur.
- Pour un exit de sécurité sur un canal MQCHT\_SVRCONN, cette zone peut contenir un identificateur utilisateur qui a été obtenu à partir de l'environnement sur le client, s'il n'y a pas d'exit de sécurité client. L'exit peut valider cet ID utilisateur (éventuellement avec le mot de passe dans *RemotePassword*) et mettre à jour la valeur dans *MCAUserIdentifier*.

S'il existe un exit de sécurité sur le client, ces informations peuvent être obtenues dans un flux de sécurité à partir du client.

La longueur de cette zone est indiquée par MQ\_USER\_ID\_LENGTH. Cette zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_2.

#### *SecurityExit (MQCHARn)*

Cette zone indique le nom de l'exit de sécurité du canal.

Si ce nom n'est pas vide, l'exit est appelé aux moments suivants:

- Immédiatement après l'établissement d'un canal.

Avant que les messages soient transférés, l'exit a la possibilité d'émettre un flux de messages de sécurisation afin de valider des droits de connexion.

- A la réception d'une réponse à un flux de messages de sécurité.

Tous les flux de messages de sécurité reçus du processeur distant sur la machine distante sont transmis à l'exit.

- Lors de l'initialisation et de l'arrêt du canal.

Voir «MQCD-Définition de canal», à la page 1553 pour une description du contenu de cette zone dans divers environnements.

La longueur de cette zone est indiquée par MQ\_EXIT\_NAME\_LENGTH.

**Remarque :** La valeur de cette constante est spécifique à l'environnement.

#### *Données SecurityUser(MQCHAR32)*

Ce canal indique les données utilisateur de l'exit de sécurité du canal.

Ces données sont transmises à l'exit de sécurité de canal dans la zone *ExitData* du paramètre **ChannelExitParms** (voir MQ\_CHANNEL\_EXIT).

Cette zone contient initialement les données qui ont été définies dans la définition de canal. Toutefois, pendant la durée de vie de cette instance MCA, toutes les modifications apportées au contenu de cette zone par un exit de n'importe quel type sont conservées par l'agent MCA et rendues visibles par les appels ultérieurs des exits (quel que soit le type) pour cette instance MCA. Cela s'applique aux exits sur différentes conversations. Ces modifications n'affectent pas la définition de canal utilisée par d'autres instances MCA. Tous les caractères (y compris les données binaires) peuvent être utilisés.

La longueur de cette zone est donnée par MQ\_EXIT\_DATA\_LENGTH.

Cette zone n'est pas pertinente dans IBM MQ for IBM i.

#### *SendExit (MQCHARn)*

Cette zone indique le nom de l'exit d'émission de canal.

Si ce nom n'est pas vide, l'exit est appelé aux moments suivants:

- Immédiatement avant l'envoi des données sur le réseau.

L'exit reçoit la mémoire tampon de transmission complète avant qu'elle ne soit transmise. Le contenu de la mémoire tampon peut être modifié selon les besoins.

- Lors de l'initialisation et de l'arrêt du canal.

Voir «MQCD-Définition de canal», à la page 1553 pour une description du contenu de cette zone dans divers environnements.

La longueur de cette zone est indiquée par MQ\_EXIT\_NAME\_LENGTH.

**Remarque :** La valeur de cette constante est spécifique à l'environnement.

#### *SendExitPtr (MQPTR)*

Cette zone indique l'adresse de la première zone *SendExit*.

Si *SendExitsDefined* est supérieur à zéro, cette adresse est l'adresse de la liste des noms de chaque exit d'émission de canal de la chaîne.

Chaque nom se trouve dans une zone de longueur *ExitNameLength*, complétée à droite par des blancs. Il y a des champs *SendExitsDefined* adjacents les uns aux autres-un pour chaque sortie.

Toutes les modifications apportées à ces noms par un exit sont conservées, bien que l'exit d'envoi de message n'effectue aucune action explicite-il ne modifie pas les exits qui sont appelés.

Si *SendExitsDefined* est égal à zéro, cette zone est le pointeur null.

Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_4.

#### *SendExitsDéfini (MQLONG)*

Cette zone indique le nombre d'exits d'émission de canal définis dans la chaîne.

Elle est supérieure ou égale à zéro.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_4.

#### *Données SendUser(MQCHAR32)*

Cette zone indique les données utilisateur de l'exit d'émission de canal.

Ces données sont transmises à l'exit d'émission de canal dans la zone *ExitData* du paramètre **ChannelExitParms** (voir MQ\_CHANNEL\_EXIT).

Cette zone contient initialement les données qui ont été définies dans la définition de canal. Toutefois, pendant la durée de vie de cette instance MCA, toutes les modifications apportées au contenu de cette zone par un exit de n'importe quel type sont conservées par l'agent MCA et rendues visibles par les appels ultérieurs des exits (quel que soit le type) pour cette instance MCA. Cela s'applique aux exits sur différentes conversations. Ces modifications n'affectent pas la définition de canal utilisée par d'autres instances MCA. Tous les caractères (y compris les données binaires) peuvent être utilisés.

La longueur de cette zone est donnée par MQ\_EXIT\_DATA\_LENGTH.

Cette zone n'est pas pertinente dans IBM MQ for IBM i.

#### *SendUserDataPtr (MQPTR)*

Cette zone indique l'adresse de la zone *SendUserData*.

Si *SendExitsDefined* est supérieur à zéro, cette adresse correspond à l'adresse de la liste des éléments de données utilisateur pour chaque exit de message de canal de la chaîne.

Chaque élément de données utilisateur se trouve dans une zone de longueur *ExitDataLength*, complétée à droite par des blancs. Il y a des champs *MsgExitsDefined* adjacents les uns aux autres-un pour chaque sortie. Si le nombre d'éléments de données utilisateur définis est inférieur au nombre de noms d'exit, les éléments de données utilisateur non définis sont mis à blanc. À l'inverse, si le nombre d'éléments de données utilisateur définis est supérieur au nombre de noms d'exit, les éléments de données utilisateur excédentaires sont ignorés et ne sont pas présentés à l'exit.

Toutes les modifications apportées à ces valeurs par un exit sont conservées. Cela permet à un exit de transmettre des informations à un autre exit. Aucune validation n'est effectuée sur les modifications. Par exemple, des données binaires peuvent être écrites dans ces zones si nécessaire.

Si *SendExitsDefined* est égal à zéro, cette zone est le pointeur null.

Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_4.



#### *SeqNumberRetour à la ligne (MQLONG)*

Cette zone indique le numéro de séquence de message le plus élevé autorisé.

Lorsque cette valeur est atteinte, le retour à la ligne des numéros de séquence est redémarré à 1.

Cette valeur n'est pas négociable et doit correspondre aux définitions de canal local et éloigné.

Cette zone n'est pas pertinente pour les canaux dont le *ChannelType* est MQCHT\_SVRCONN ou MQCHT\_CLNTCONN.

#### *SharingConversations (MQLONG)*

Cette zone indique le nombre maximal de conversations pouvant partager une instance de canal associée à ce canal.

Cette zone est utilisée sur les canaux de connexion client et de connexion serveur.

La valeur 0 signifie que le canal fonctionne comme dans les versions antérieures à IBM WebSphere MQ 7.0 en ce qui concerne les attributs suivants:

- Partage de conversation
- Lecture anticipée
- STOP CHANNEL(*channelname*) MODE(QUIESCE)
- Intervalle des pulsations par lots
- Consommation asynchrone du client

La valeur 1 est la valeur minimale pour le comportement de IBM MQ . Bien qu'une seule conversation soit autorisée sur l'instance de canal, la lecture anticipée, la consommation asynchrone et le comportement de la pulsation CLNTCONN-SVRCONN et de l'arrêt du canal au repos sont disponibles.

Il s'agit d'une zone d'entrée de l'exit. Elle n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_9.

La valeur par défaut de cette zone est 10.

**Remarque :** Les limites *MaxInstances* et *MaxInstancesPerClient* appliquées à un canal limitent le nombre d'instances de canal et non le nombre de conversations pouvant partager ces instances.

#### *Nom ShortConnection(MQCHAR20)*

Cette zone indique les 20 premiers octets d'un nom de connexion.

Si la zone *Version* est MQCD\_VERSION\_1, *ShortConnectionName* contient le nom de connexion complet.

Si la zone *Version* est MQCD\_VERSION\_2 ou supérieur, *ShortConnectionName* contient les 20 premiers caractères du nom de connexion. Le nom de connexion complet est donné par la zone *ConnectionName* ; *ShortConnectionName* et les 20 premiers caractères de *ConnectionName* sont identiques.

Pour plus de détails sur le contenu de cette zone, voir *ConnectionName* .

**Remarque :** Le nom de cette zone a été modifié pour MQCD\_VERSION\_2 et les versions ultérieures de MQCD ; la zone était auparavant appelée *ConnectionName*.

La longueur de cette zone est indiquée par MQ\_SHORT\_CONN\_NAME\_LENGTH.

#### *Nombre de ShortRetry(MQLONG)*

Cette zone indique le nombre maximal de tentatives de connexion à une machine distante.

Cette zone indique le nombre maximal de tentatives de connexion à la machine distante, à des intervalles définis par *ShortRetryInterval*, avant que les valeurs (normalement plus longues) *LongRetryCount* et *LongRetryInterval* ne soient utilisées.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_CLUSSDR ou MQCHT\_CLUSRCVR.

### *Intervalle ShortRetry(MQLONG)*

Cette zone indique le nombre maximal de secondes à attendre avant de tenter à nouveau la connexion à la machine distante.

L'intervalle entre les nouvelles tentatives peut être étendu si le canal doit attendre pour devenir actif.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_CLUSSDR ou MQCHT\_CLUSRCVR.

### *SPLProtection (MQLONG)*

Cette zone indique la valeur de la protection des règles de sécurité AMS .

La valeur est l'une des suivantes :

#### **MQSPL\_PASSTHRU**

Transmettez, sans modification, tous les messages envoyés ou reçus par l'agent MCA pour ce canal.

Cette valeur est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_RECEIVER ou MQCHT\_REQUESTER, et correspond à la valeur par défaut.

#### **MQSPL\_REMOVE**

Supprimez toute protection AMS des messages extraits de la file d'attente de transmission par l'agent MCA et envoyez les messages au partenaire.

Cette valeur est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_SENDER ou MQCHT\_SERVER.

#### **MQSPL\_ASPOLICY**

En fonction de la stratégie définie pour la file d'attente cible, application de la protection AMS aux messages entrants avant leur placement dans la file d'attente cible.

Cette valeur est pertinente uniquement pour les canaux avec un *ChannelType* de MQCHT\_RECEIVER ou MQCHT\_REQUESTER.

Il s'agit d'une zone d'entrée de l'exit. Cette zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_12.

### *SSLCipherSpec (MQCHAR32)*

Cette zone indique la spécification de chiffrement utilisée lors de l'utilisation de TLS.

Si SSLCipherSpec est vide, le canal n'utilise pas TLS. S'il n'est pas vide, cette zone contient une chaîne spécifiant le CipherSpec utilisé.

Ce paramètre est valide pour tous les types de canal. Il est pris en charge sur les plateformes suivantes:

-  AIX
-  IBM i
-  Linux
-  Windows
-  z/OS

Elle est valide uniquement pour les types de canal d'un type de transport (TRPTYPE) TCP.

Il s'agit d'une zone d'entrée de l'exit. La longueur de cette zone est indiquée par MQ\_SSL\_CIPHER\_SPEC\_LENGTH. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_7.

### *SSLClientAuth (MQLONG)*

Cette zone indique si l'authentification de client TLS est requise.

Cette zone concerne uniquement les définitions de canal SVRCONN.

Ses valeurs sont l'une des suivantes :

## **MQSCA\_REQUIS**

Authentification du client requise.

## **MQSCA\_XX\_ENCODE\_CASE\_ONE facultatif**

Authentification client facultative.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_7.

### *SSLPeerNameLongueur (MQLONG)*

Cette zone indique la longueur en octets du nom d'homologue TLS désigné par *SSLPeerNamePtr*.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_7.

### *SSLPeerNamePtr (MQPTR)*

Cette zone indique l'adresse du nom d'homologue TLS.

Lorsqu'un certificat est reçu lors d'un établissement de liaison TLS réussi, le nom distinctif du sujet du certificat est copié dans la zone MQCD accessible par le Ptr *SSLPeerName* à la fin du canal qui reçoit le certificat. Elle remplace la valeur *SSLPeerName* du canal si cette valeur est présente dans la définition de canal de l'utilisateur local. Si un exit de sécurité est spécifié à cette extrémité du canal, il reçoit le nom distinctif du certificat homologue dans le MQCD.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_7.

**Remarque :** Les applications d'exit de sécurité construites avant l'édition de IBM WebSphere MQ 7.1 peuvent nécessiter une mise à jour. Pour plus d'informations, voir [Programmes d'exit de sécurité de canal](#).

### *StrucLength (MQLONG)*

Cette zone indique la longueur en octets de la structure MQCD.

La longueur n'inclut aucune des chaînes adressées par les zones de pointeur contenues dans la structure. La valeur est l'une des suivantes :

### **MQCD\_LENGTH\_4**

Longueur de la structure de définition de canal version-4 .

### **MQCD\_LENGTH\_5**

Longueur de la structure de définition de canal version-5 .

### **MQCD\_LENGTH\_6**

Longueur de la structure de définition de canal version-6 .

### **MQCD\_LENGTH\_7**

Longueur de la structure de définition de canal version-7 .

### **MQCD\_LENGTH\_8**

Longueur de la structure de définition de canal version-8 .

### **MQCD\_LENGTH\_9**

Longueur de la structure de définition de canal version-9 .

### **MQCD\_LENGTH\_10**

Longueur de la structure de définition de canal version-10 .

### **MQCD\_LENGTH\_11**

Longueur de la structure de définition de canal version-11 .

### **MQCD\_LENGTH\_12**

Longueur de la structure de définition de canal version-12 .

La constante suivante indique la longueur de la version en cours:

### **MQCD\_LONGUEUR\_EN\_COURS**

Longueur de la version en cours de la structure de définition de canal.

**Remarque :** Ces constantes ont des valeurs spécifiques à l'environnement.

La zone n'est pas présente si *Version* est inférieur à MQCD\_VERSION\_4.

*TpName (MQCHAR64)*

Cette zone indique le nom du programme de transaction LU 6.2 .

Cette zone est pertinente uniquement si le protocole de transmission (*TransportType*) est MQXPT\_LU62 et que *ChannelType* n'est pas MQCHT\_SVRCONN ou MQCHT\_RECEIVER.

Cette zone est toujours à blanc sur les plateformes sur lesquelles les informations sont contenues dans l'objet côté communications.

La longueur de cette zone est indiquée par MQ\_TP\_NAME\_LENGTH.

*TransportType (MQLONG)*

Cette zone indique le protocole de transmission à utiliser.

La valeur n'est pas vérifiée si le canal a été lancé à partir de l'autre extrémité.

Ses valeurs sont l'une des suivantes :

**MQXPT\_LU62**

Protocole de transport LU 6.2 .

**MQXPT\_TCP**

Protocole de transport TCP/IP.

**MQXPT\_NETBIOS**

Protocole de transport NetBIOS .

Cette valeur est prise en charge dans les environnements suivants: Windows.

**MQXPT\_SPX**

Protocole de transport SPX.

Cette valeur est prise en charge dans les environnements suivants: Windows, plus les clients IBM MQ connectés à ces systèmes.

*UseDLQ (MQLONG)*

Cette zone indique si la file d'attente de rebut (ou la file d'attente de messages non livrés) est utilisée lorsque les messages ne peuvent pas être distribués par les canaux.

Il peut contenir l'une des valeurs suivantes:

**MQUSEDLQ\_NO**

Les messages qui ne peuvent pas être distribués par un canal sont traités comme un échec. Le canal supprime le message ou se termine conformément au paramètre NPMSPEED.

**MQUSEDLQ\_OUI**

Lorsque l'attribut de gestionnaire de files d'attente DEADQ fournit le nom d'une file d'attente de rebut, il est utilisé, sinon le comportement est NO. YES est la valeur par défaut.

*UserIdentifier (MQCHAR12)*

Cette zone indique l'ID utilisateur utilisé par l'agent MCA lors de la tentative d'ouverture d'une session SNA sécurisée avec un agent MCA éloigné.

Cette zone ne peut être vide que sous AIX, Linux, and Windowset n'est pertinente que pour les canaux dont le *ChannelType* est MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_REQUESTER ou MQCHT\_CLNTCONN. Sous z/OS, cette zone n'est pas pertinente.

La longueur de cette zone est indiquée par MQ\_USER\_ID\_LENGTH. Cependant, seuls les 10 premiers caractères sont utilisés.

Cette zone n'est pas présente lorsque *Version* est inférieur à MQCD\_VERSION\_2.

*Version (MQLONG)*

La zone *Version* indique le numéro de version le plus élevé que vous pouvez définir pour la structure.

La valeur dépend de l'environnement:

**MQCD \_VERSION\_1**

Structure de définition de canal version 1.

**MQCD \_VERSION\_2**

Structure de définition de canal version 2.

**MQCD \_VERSION\_3**

Structure de définition de canal version 3.

**MQCD \_VERSION\_4**

Structure de définition de canal version 4.

**MQCD \_VERSION\_5**

Structure de définition de canal version 5.

**MQCD \_VERSION\_6**

Structure de définition de canal version 6.

**MQCD \_VERSION\_7**

Structure de définition de canal version 7.

**MQCD \_VERSION\_8**

Structure de définition de canal version 8.

**MQCD \_VERSION\_9**

Structure de définition de canal version 9.

**MQCD \_VERSION\_10**

Structure de définition de canal version 10.

**MQCD \_VERSION\_11**

Structure de définition de canal version 11.

La version 11 est la valeur la plus élevée que vous pouvez définir pour la zone sur IBM MQ 8.0 sur toutes les plateformes.

 **MQCD \_VERSION\_12**

Structure de définition de canal version 12.

La version 12 est la valeur la plus élevée que vous pouvez définir pour la zone sur IBM MQ 9.1.3.

Les zones qui existent uniquement dans les versions les plus récentes de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

**MQCD\_CURRENT\_VERSION**

La valeur définie dans `MQCD_CURRENT_VERSION` est la version actuelle de la structure de définition de canal utilisée.

La valeur de `MQCD_CURRENT_VERSION` dépend de l'environnement. Il contient la valeur la plus élevée prise en charge par la plateforme.

`MQCD_CURRENT_VERSION` n'est pas utilisé pour initialiser les structures par défaut fournies dans les fichiers d'en-tête, de copie et d'inclusion fournis pour les différents langages de programmation. L'initialisation par défaut de *Version* dépend de la plateforme et de l'édition.

Les déclarations `MQCD` dans les fichiers d'en-tête, de copie et d'inclusion sont initialisées dans `MQCD_VERSION_6`. Pour utiliser des zones `MQCD` supplémentaires, les applications doivent définir le numéro de version sur `MQCD_CURRENT_VERSION`. Si vous écrivez une application portable entre plusieurs environnements, vous devez choisir une version prise en charge dans tous les environnements.

**Conseil :** Lorsqu'une nouvelle version de la structure MQCD est introduite, la présentation de la partie existante n'est pas modifiée. L'exit doit vérifier le numéro de version. Elle doit être supérieure ou égale à la version la plus basse qui contient les zones que l'exit doit utiliser.

#### *XmitQName (MQCHAR48)*

Cette zone indique le nom de la file d'attente de transmission à partir de laquelle les messages sont extraits.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT\_SENDER ou MQCHT\_SERVER.

La longueur de cette zone est indiquée par MQ\_Q\_NAME\_LENGTH.

### **Déclaration C**

Cette déclaration est la déclaration C pour la structure MQCD.

```
typedef struct tagMQCD MQCD;
typedef MQCD MQPOINTER PMQCD;
typedef PMQCD MQPOINTER PPMQCD;

struct tagMQCD {
    MQCHAR    ChannelName[20];           /* Channel definition name */
    MQLONG    Version;                  /* Structure version number */
    MQLONG    ChannelType;              /* Channel type */
    MQLONG    TransportType;            /* Transport type */
    MQCHAR    Desc[64];                  /* Channel description */
    MQCHAR    QMgrName[48];              /* Queue manager name */
    MQCHAR    XmitQName[48];            /* Transmission queue name */
    MQCHAR    ShortConnectionName[20];  /* First 20 bytes of */
                                          /* connection name */
    MQCHAR    MCAName[20];               /* Reserved */
    MQCHAR    ModeName[8];               /* LU 6.2 Mode name */
    MQCHAR    TpName[64];               /* LU 6.2 transaction program */
                                          /* name */
    MQLONG    BatchSize;                 /* Batch size */
    MQLONG    DiscInterval;              /* Disconnect interval */
    MQLONG    ShortRetryCount;           /* Short retry count */
    MQLONG    ShortRetryInterval;        /* Short retry wait interval */
    MQLONG    LongRetryCount;            /* Long retry count */
    MQLONG    LongRetryInterval;         /* Long retry wait interval */
    MQCHAR    SecurityExit[128];         /* Channel security exit name */
    MQCHAR    MsgExit[128];              /* Channel message exit name */
    MQCHAR    SendExit[128];             /* Channel send exit name */
    MQCHAR    ReceiveExit[128];          /* Channel receive exit name */
    MQLONG    SeqNumberWrap;             /* Highest allowable message */
                                          /* sequence number */
    MQLONG    MaxMsgLength;              /* Maximum message length */
    MQLONG    PutAuthority;               /* Put authority */
    MQLONG    DataConversion;            /* Data conversion */
    MQCHAR    SecurityUserData[32];       /* Channel security exit user */
                                          /* data */
    MQCHAR    MsgUserData[32];           /* Channel message exit user */
                                          /* data */
    MQCHAR    SendUserData[32];          /* Channel send exit user */
                                          /* data */
    MQCHAR    ReceiveUserData[32];        /* Channel receive exit user */
                                          /* data */
    /* Ver:1 */
    MQCHAR    UserIdentifier[12];         /* User identifier */
    MQCHAR    Password[12];              /* Password */
    MQCHAR    MCAUserIdentifier[12];     /* First 12 bytes of MCA user */
                                          /* identifier */
    MQLONG    MCAType;                    /* Message channel agent type */
    MQCHAR    ConnectionName[264];       /* Connection name */
    MQCHAR    RemoteUserIdentifier[12];  /* First 12 bytes of user */
                                          /* identifier from partner */
    MQCHAR    RemotePassword[12];        /* Password from partner */
    /* Ver:2 */
    MQCHAR    MsgRetryExit[128];         /* Channel message retry exit */
                                          /* name */
    MQCHAR    MsgRetryUserData[32];       /* Channel message retry exit */
                                          /* user data */
    MQLONG    MsgRetryCount;              /* Number of times MCA will */
                                          /* try to put the message, */
                                          /* after first attempt has */
                                          /* failed */
}
```

```

MQLONG     MsgRetryInterval;           /* Minimum interval in */
                                                /* milliseconds after which */
                                                /* the open or put operation */
                                                /* will be retried */

/* Ver:3 */
MQLONG     HeartbeatInterval;         /* Time in seconds between */
                                                /* heartbeat flows */
MQLONG     BatchInterval;             /* Batch duration */
MQLONG     NonPersistentMsgSpeed;     /* Speed at which */
                                                /* nonpersistent messages are */
                                                /* sent */
MQLONG     StrucLength;               /* Length of MQCD structure */
MQLONG     ExitNameLength;            /* Length of exit name */
MQLONG     ExitDataLength;            /* Length of exit user data */
MQLONG     MsgExitsDefined;           /* Number of message exits */
                                                /* defined */
MQLONG     SendExitsDefined;          /* Number of send exits */
                                                /* defined */
MQLONG     ReceiveExitsDefined;       /* Number of receive exits */
                                                /* defined */
MQPTR      MsgExitPtr;                /* Address of first MsgExit */
                                                /* field */
MQPTR      MsgUserDataPtr;            /* Address of first */
                                                /* MsgUserData field */
MQPTR      SendExitPtr;               /* Address of first SendExit */
                                                /* field */
MQPTR      SendUserDataPtr;           /* Address of first */
                                                /* SendUserData field */
MQPTR      ReceiveExitPtr;            /* Address of first */
                                                /* ReceiveExit field */
MQPTR      ReceiveUserDataPtr;        /* Address of first */
                                                /* ReceiveUserData field */

/* Ver:4 */
MQPTR      ClusterPtr;                /* Address of a list of */
                                                /* cluster names */
MQLONG     ClustersDefined;           /* Number of clusters to */
                                                /* which the channel belongs */
MQLONG     NetworkPriority;           /* Network priority */

/* Ver:5 */
MQLONG     LongMCAUserIdLength;       /* Length of long MCA user */
                                                /* identifier */
MQLONG     LongRemoteUserIdLength;   /* Length of long remote user */
                                                /* identifier */
MQPTR      LongMCAUserIdPtr;          /* Address of long MCA user */
                                                /* identifier */
MQPTR      LongRemoteUserIdPtr;       /* Address of long remote */
                                                /* user identifier */
MQBYTE40   MCASecurityId;             /* MCA security identifier */
MQBYTE40   RemoteSecurityId;         /* Remote security identifier */

/* Ver:6 */
MQCHAR     SSLCipherSpec[32];         /* TLS CipherSpec */
MQPTR      SSLPeerNamePtr;           /* Address of TLS peer name */
MQLONG     SSLPeerNameLength;        /* Length of TLS peer name */
MQLONG     SSLClientAuth;            /* Whether TLS client */
                                                /* authentication is required */
MQLONG     KeepAliveInterval;         /* Keepalive interval */
MQCHAR     LocalAddress[48];          /* Local communications */
                                                /* address */
MQLONG     BatchHeartbeat;            /* Batch heartbeat interval */

/* Ver:7 */
MQLONG     HdrCompList[2];            /* Header data compression */
                                                /* list */
MQLONG     MsgCompList[16];          /* Message data compression */
                                                /* list */
MQLONG     CLWLChannelRank;           /* Channel rank */
MQLONG     CLWLChannelPriority;        /* Channel priority */
MQLONG     CLWLChannelWeight;         /* Channel weight */
MQLONG     ChannelMonitoring;         /* Channel monitoring */
MQLONG     ChannelStatistics;         /* Channel statistics */

/* Ver:8 */
MQLONG     SharingConversations;      /* Limit on sharing */
                                                /* conversations */
MQLONG     PropertyControl;           /* Message property control */
MQLONG     MaxInstances;              /* Limit on SVRCONN channel */
                                                /* instances */
MQLONG     MaxInstancesPerClient;     /* Limit on SVRCONN channel */
                                                /* instances per client */
MQLONG     ClientChannelWeight;       /* Client channel weight */
MQLONG     ConnectionAffinity;        /* Connection affinity */

/* Ver:9 */
MQLONG     BatchDataLimit;            /* Batch data limit */
MQLONG     UseDLQ;                   /* Use Dead Letter Queue */

```

```

MQLONG    DefReconnect;                /* Default client reconnect */
                                                /* option */
/* Ver:10 */
MQCHAR64  CertificateLabel;            /* Certificate label */
/* Ver:11 */
MQLONG    SPLProtection                /* AMS Security policy protection */
/* Ver:12 */
};

```

## Déclaration COBOL

Cette déclaration est la déclaration COBOL pour la structure MQCD.

```

** MQCD structure
   10 MQCD.
      ** Channel definition name
         15 MQCD-CHANNELNAME PIC X(20).
      ** Structure version number
         15 MQCD-VERSION PIC S9(9) BINARY.
      ** Channel type
         15 MQCD-CHANNELTYPE PIC S9(9) BINARY.
      ** Transport type
         15 MQCD-TRANSPORTTYPE PIC S9(9) BINARY.
      ** Channel description
         15 MQCD-DESC PIC X(64).
      ** Queue manager name
         15 MQCD-QMGRNAME PIC X(48).
      ** Transmission queue name
         15 MQCD-XMITQNAME PIC X(48).
      ** First 20 bytes of connection name
         15 MQCD-SHORTCONNECTIONNAME PIC X(20).
      ** Reserved
         15 MQCD-MCANAME PIC X(20).
      ** LU 6.2 Mode name
         15 MQCD-MODENAME PIC X(8).
      ** LU 6.2 transaction program name
         15 MQCD-TPNAME PIC X(64).
      ** Batch size
         15 MQCD-BATCHSIZE PIC S9(9) BINARY.
      ** Disconnect interval
         15 MQCD-DISCINTERVAL PIC S9(9) BINARY.
      ** Short retry count
         15 MQCD-SHORTRETRYCOUNT PIC S9(9) BINARY.
      ** Short retry wait interval
         15 MQCD-SHORTRETRYINTERVAL PIC S9(9) BINARY.
      ** Long retry count
         15 MQCD-LONGRETRYCOUNT PIC S9(9) BINARY.
      ** Long retry wait interval
         15 MQCD-LONGRETRYINTERVAL PIC S9(9) BINARY.
      ** Channel security exit name
         15 MQCD-SECURITYEXIT PIC X(20).
      ** Channel message exit name
         15 MQCD-MSGEXIT PIC X(20).
      ** Channel send exit name
         15 MQCD-SENDEXIT PIC X(20).
      ** Channel receive exit name
         15 MQCD-RECEIVEEXIT PIC X(20).
      ** Highest allowable message sequence number
         15 MQCD-SEQNUMBERWRAP PIC S9(9) BINARY.
      ** Maximum message length
         15 MQCD-MAXMSGLENGTH PIC S9(9) BINARY.
      ** Put authority
         15 MQCD-PUTAUTHORITY PIC S9(9) BINARY.
      ** Data conversion
         15 MQCD-DATACONVERSION PIC S9(9) BINARY.
      ** Channel security exit user data
         15 MQCD-SECURITYUSERDATA PIC X(32).
      ** Channel message exit user data
         15 MQCD-MSGUSERDATA PIC X(32).
      ** Channel send exit user data
         15 MQCD-SENDUSERDATA PIC X(32).
      ** Channel receive exit user data
         15 MQCD-RECEIVEUSERDATA PIC X(32).
      ** Ver:1 **
      ** User identifier
         15 MQCD-USERIDENTIFIER PIC X(12).
      ** Password
         15 MQCD-PASSWORD PIC X(12).
      ** First 12 bytes of MCA user identifier

```



```

15 MQCD-MCAUSERIDENTIFIER PIC X(12).
** Message channel agent type
15 MQCD-MCATYPE PIC S9(9) BINARY.
** Connection name
15 MQCD-CONNECTIONNAME PIC X(264).
** First 12 bytes of user identifier from partner
15 MQCD-REMOTEUSERIDENTIFIER PIC X(12).
** Password from partner
15 MQCD-REMOTEPASSWORD PIC X(12).
** Ver:2 **
** Channel message retry exit name
15 MQCD-MSGRETRYEXIT PIC X(20).
** Channel message retry exit user data
15 MQCD-MSGRETRYUSERDATA PIC X(32).
** Number of times MCA will try to put the message, after first
** attempt has failed
15 MQCD-MSGRETRYCOUNT PIC S9(9) BINARY.
** Minimum interval in milliseconds after which the open or put
** operation will be retried
15 MQCD-MSGRETRYINTERVAL PIC S9(9) BINARY.
** Ver:3 **
** Time in seconds between heartbeat flows
15 MQCD-HEARTBEATINTERVAL PIC S9(9) BINARY.
** Batch duration
15 MQCD-BATCHINTERVAL PIC S9(9) BINARY.
** Speed at which nonpersistent messages are sent
15 MQCD-NONPERSISTENTMSGSPPEED PIC S9(9) BINARY.
** Length of MQCD structure
15 MQCD-STRUCLLENGTH PIC S9(9) BINARY.
** Length of exit name
15 MQCD-EXITNAMELENGTH PIC S9(9) BINARY.
** Length of exit user data
15 MQCD-EXITDATALENGTH PIC S9(9) BINARY.
** Number of message exits defined
15 MQCD-MSGEXITSDEFINED PIC S9(9) BINARY.
** Number of send exits defined
15 MQCD-SENDEXITSDEFINED PIC S9(9) BINARY.
** Number of receive exits defined
15 MQCD-RECEIVEEXITSDEFINED PIC S9(9) BINARY.
** Address of first MsgExit field
15 MQCD-MSGEXITPTR POINTER.
** Address of first MsgUserData field
15 MQCD-MSGUSERDATAPTR POINTER.
** Address of first SendExit field
15 MQCD-SENDEXITPTR POINTER.
** Address of first SendUserData field
15 MQCD-SENDUSERDATAPTR POINTER.
** Address of first ReceiveExit field
15 MQCD-RECEIVEEXITPTR POINTER.
** Address of first ReceiveUserData field
15 MQCD-RECEIVEUSERDATAPTR POINTER.
** Ver:4 **
** Address of a list of cluster names
15 MQCD-CLUSTERPTR POINTER.
** Number of clusters to which the channel belongs
15 MQCD-CLUSTERSDEFINED PIC S9(9) BINARY.
** Network priority
15 MQCD-NETWORKPRIORITY PIC S9(9) BINARY.
** Ver:5 **
** Length of long MCA user identifier
15 MQCD-LONGMCAUSERIDLENGTH PIC S9(9) BINARY.
** Length of long remote user identifier
15 MQCD-LONGREMOTEUSERIDLENGTH PIC S9(9) BINARY.
** Address of long MCA user identifier
15 MQCD-LONGMCAUSERIDPTR POINTER.
** Address of long remote user identifier
15 MQCD-LONGREMOTEUSERIDPTR POINTER.
** MCA security identifier
15 MQCD-MCASECURITYID PIC X(40).
** Remote security identifier
15 MQCD-REMOTESESECURITYID PIC X(40).
** Ver:6 **
** TLS CipherSpec
15 MQCD-SSLCIPHERSPEC PIC X(32).
** Address of TLS peer name
15 MQCD-SSLPEERNAMEPTR POINTER.
** Length of TLS peer name
15 MQCD-SSLPEERNAMELENGTH PIC S9(9) BINARY.
** Whether TLS client authentication is required
15 MQCD-SSLCLIENTAUTH PIC S9(9) BINARY.
** Keepalive interval
15 MQCD-KEEPALIVEINTERVAL PIC S9(9) BINARY.

```

```

** Local communications address
  15 MQCD-LOCALADDRESS PIC X(48).
** Batch heartbeat interval
  15 MQCD-BATCHHEARTBEAT PIC S9(9) BINARY.
** Ver:7 **
** Header data compression list
  15 MQCD-HDRCOMPLIST PIC S9(9) BINARY.
** Message data compression list
  15 MQCD-MSGCOMPLIST PIC S9(9) BINARY.
** Channel rank
  15 MQCD-CLWLCHANNELRANK PIC S9(9) BINARY.
** Channel priority
  15 MQCD-CLWLCHANNELPRIORITY PIC S9(9) BINARY.
** Channel weight
  15 MQCD-CLWLCHANNELWEIGHT PIC S9(9) BINARY.
** Channel monitoring
  15 MQCD-CHANNELMONITORING PIC S9(9) BINARY.
** Channel statistics
  15 MQCD-CHANNELSTATISTICS PIC S9(9) BINARY.
** Ver:8 **
** Limit on sharing conversations
  15 MQCD-SHARINGCONVERSATIONS PIC S9(9) BINARY.
** Message property control
  15 MQCD-PROPERTYCONTROL PIC S9(9) BINARY.
** Limit on SVRCONN channel instances
  15 MQCD-MAXINSTANCES PIC S9(9) BINARY.
** Limit on SVRCONN channel instances per client
  15 MQCD-MAXINSTANCESPERCLIENT PIC S9(9) BINARY.
** Client channel weight
  15 MQCD-CLIENTCHANNELWEIGHT PIC S9(9) BINARY.
** Connection affinity
  15 MQCD-CONNECTIONAFFINITY PIC S9(9) BINARY.
** Ver:9 **
** Batch data limit
  15 MQCD-BATCHDATALIMIT PIC S9(9) BINARY.
** Use Dead Letter Queue
  15 MQCD-USEDLQ PIC S9(9) BINARY.
** Default client reconnect option
  15 MQCD-DEFRECONNECT PIC S9(9) BINARY.
** Ver:10 **
** Certificate Label
  15 MQCD-CERTLABL PIC X (64)
** Ver:11 **
** AMS Security policy protection
  15 MQCD-SPLPROTECTION PIC S9(9) BINARY
** Ver:12 **

```

### **Déclaration RPG (ILE)**

Cette déclaration est la déclaration RPG pour la structure MQCD.

```

D* MQCD Structure
D*
D* Channel definition name
D CDCHN          1      20
D* Structure version number
D CDVER          21     24I 0
D* Channel type
D CDCHT          25     28I 0
D* Transport type
D CDTRT          29     32I 0
D* Channel description
D CDDDES         33     96
D* Queue manager name
D CDQM           97     144
D* Transmission queue name
D CDXQ          145     192
D* First 20 bytes of connection name
D CDSCN         193     212
D* Reserved
D CDMCA         213     232
D* LU 6.2 Mode name
D CDMOD         233     240
D* LU 6.2 transaction program name
D CDTP          241     304
D* Batch size
D CDBS          305     308I 0
D* Disconnect interval
D CDDI          309     312I 0

```

```

D* Short retry count
D CDSRC          313    316I 0
D* Short retry wait interval
D CDSRI          317    320I 0
D* Long retry count
D CDLRC          321    324I 0
D* Long retry wait interval
D CDLRI          325    328I 0
D* Channel security exit name
D CDSCX          329    348
D* Channel message exit name
D CDMSX          349    368
D* Channel send exit name
D CDSNX          369    388
D* Channel receive exit name
D CDRCX          389    408
D* Highest allowable message sequence number
D CDSNW          409    412I 0
D* Maximum message length
D CDMML          413    416I 0
D* Put authority
D CDPA           417    420I 0
D* Data conversion
D CDDC           421    424I 0
D* Channel security exit user data
D CDSCD          425    456
D* Channel message exit user data
D CDMSD          457    488
D* Channel send exit user data
D CDSND          489    520
D* Channel receive exit user data
D CDRCU          521    552
D* Ver:1 **
D* User identifier
D CDUID          553    564
D* Password
D CDPW           565    576
D* First 12 bytes of MCA user identifier
D CDAUI          577    588
D* Message channel agent type
D CDCAT          589    592I 0
D* Connection name
D CDCON          593    848
D CDCN2          849    856
D* First 12 bytes of user identifier from partner
D CDRUI          857    868
D* Password from partner
D CDRPW          869    880
D* Ver:2 **
D* Channel message retry exit name
D CDMRX          881    900
D* Channel message retry exit user data
D CDMRD          901    932
D* Number of times MCA will try to put the message, after first
D* attempt has failed
D CDMRC          933    936I 0
D* Minimum interval in milliseconds after which the open or put
D* operation will be retried
D CDMRI          937    940I 0
D* Ver:3 **
D* Time in seconds between heartbeat flows
D CDHBI          941    944I 0
D* Batch duration
D CDBI           945    948I 0
D* Speed at which nonpersistent messages are sent
D CDNPM          949    952I 0
D* Length of MQCD structure
D CDLEN          953    956I 0
D* Length of exit name
D CDXNL          957    960I 0
D* Length of exit user data
D CDXDL          961    964I 0
D* Number of message exits defined
D CDMXD          965    968I 0
D* Number of send exits defined
D CDSXD          969    972I 0
D* Number of receive exits defined
D CDRXD          973    976I 0
D* Address of first MsgExit field
D CDMXP          977    992*
D* Address of first MsgUserData field
D CDMUP          993    1008*

```

```

D* Address of first SendExit field
D CDSXP          1009  1024*
D* Address of first SendUserData field
D CDSUP          1025  1040*
D* Address of first ReceiveExit field
D CDRXP          1041  1056*
D* Address of first ReceiveUserData field
D CDRUP          1057  1072*
D* Ver:4 **
D* Address of a list of cluster names
D CDCLP          1073  1088*
D* Number of clusters to which the channel belongs
D CDCLD          1089  1092I 0
D* Network priority
D CDNP           1093  1096I 0
D* Ver:5 **
D* Length of long MCA user identifier
D CDMLL          1097  1100I 0
D* Length of long remote user identifier
D CDRL           1101  1104I 0
D* Address of long MCA user identifier
D CDLMP          1105  1120*
D* Address of long remote user identifier
D CDLRP          1121  1136*
D* MCA security identifier
D CDMSI          1137  1176
D* Remote security identifier
D CDRSI          1177  1216
D* Ver:6 **
D* TLS CipherSpec
D CDSCS          1217  1248
D* Address of TLS peer name
D CDSPN          1249  1264*
D* Length of TLS peer name
D CDSPL          1265  1268I 0
D* Whether TLS client authentication is required
D CDSCA          1269  1272I 0
D* Keepalive interval
D CDKAI          1273  1276I 0
D* Local communications address
D CDLOA          1277  1324
D* Batch heartbeat interval
D CDBHB          1325  1328I 0
D* Ver:7 **
D* Header data compression list
D CDHCL0
D CDHCL1          1329  1332I 0
D CDHCL2          1333  1336I 0
D CDHCL          10I 0 DIM(2) OVERLAY(CDHCL0)
D* Message data compression list
D CDMCL0
D CDMCL1          1337  1340I 0
D CDMCL2          1341  1344I 0
D CDMCL3          1345  1348I 0
D CDMCL4          1349  1352I 0
D CDMCL5          1353  1356I 0
D CDMCL6          1357  1360I 0
D CDMCL7          1361  1364I 0
D CDMCL8          1365  1368I 0
D CDMCL9          1369  1372I 0
D CDMCL10         1373  1376I 0
D CDMCL11         1377  1380I 0
D CDMCL12         1381  1384I 0
D CDMCL13         1385  1388I 0
D CDMCL14         1389  1392I 0
D CDMCL15         1393  1396I 0
D CDMCL16         1397  1400I 0
D CDMCL          10I 0 DIM(16) OVERLAY(CDMCL0)
D* Channel rank
D CDCWCR         1401  1404I 0
D* Channel priority
D CDCWCP         1405  1408I 0
D* Channel weight
D CDCWCW         1409  1412I 0
D* Channel monitoring
D CDCHLMON       1413  1416I 0
D* Channel statistics
D CDCHLST        1417  1420I 0
D* Ver:8 **
D* Limit on sharing conversations
D CDSHC          1421  1424I 0
D* Message property control

```

```

D CDPRC 1425 1428I 0
D* Limit on SVRCONN channel instances
D CDMXIN 1429 1432I 0
D* Limit on SVRCONN channel instances per client
D CDMXIC 1433 1436I 0
D* Client channel weight
D CDCLNCHLW 1437 1440I 0
D* Connection affinity
D CDCONNAFF 1441 1444I 0
D* Ver:9 **
D* Batch data limit
D CDBDL 1445 1448I 0
D* Use Dead Letter Queue
D CDUDLQ 1449 1452I 0
D* Default client reconnect option
D CDDRCN 1453 1456I 0
D* Ver:10 **

```

## Déclaration de l'assembleur System/390

Cette déclaration est la déclaration d'assembleur System/390 pour la structure MQCD.

MQCD	DSECT		
MQCD_CHANNELNAME	DS	CL20	Channel definition name
MQCD_VERSION	DS	F	Structure version number
MQCD_CHANNELTYPE	DS	F	Channel type
MQCD_TRANSPORTTYPE	DS	F	Transport type
MQCD_DESC	DS	CL64	Channel description
MQCD_QMGRNAME	DS	CL48	Queue manager name
MQCD_XMITQNAME	DS	CL48	Transmission queue name
MQCD_SHORTCONNECTIONNAME	DS	CL20	First 20 bytes of connection name
*			
MQCD_MCANAME	DS	CL20	Reserved
MQCD_MODENAME	DS	CL8	LU 6.2 Mode name
MQCD_TPNAME	DS	CL64	LU 6.2 transaction program name
MQCD_BATCHSIZE	DS	F	Batch size
MQCD_DISCINTERVAL	DS	F	Disconnect interval
MQCD_SHORTRETRYCOUNT	DS	F	Short retry count
MQCD_SHORTRETRYINTERVAL	DS	F	Short retry wait interval
MQCD_LONGRETRYCOUNT	DS	F	Long retry count
MQCD_LONGRETRYINTERVAL	DS	F	Long retry wait interval
MQCD_SECURITYEXIT	DS	CLn	Channel security exit name
MQCD_MSGEXIT	DS	CLn	Channel message exit name
MQCD_SENDEXIT	DS	CLn	Channel send exit name
MQCD_RECEIVEEXIT	DS	CLn	Channel receive exit name
MQCD_SEQUENCEWRAP	DS	F	Highest allowable message sequence number
*			
MQCD_MAXMSGLLENGTH	DS	F	Maximum message length
MQCD_PUTAUTHORITY	DS	F	Put authority
MQCD_DATAACONVERSION	DS	F	Data conversion
MQCD_SECURITYUSERDATA	DS	CL32	Channel security exit user data
MQCD_MSGUSERDATA	DS	CL32	Channel message exit user data
MQCD_SENDUSERDATA	DS	CL32	Channel send exit user data
MQCD_RECEIVEUSERDATA	DS	CL32	Channel receive exit user data
MQCD_USERIDENTIFIER	DS	CL12	User identifier
MQCD_PASSWORD	DS	CL12	Password
MQCD_MCAUSERIDENTIFIER	DS	CL12	First 12 bytes of MCA user identifier
*			
MQCD_MCATYPE	DS	F	Message channel agent type
MQCD_CONNECTIONNAME	DS	CL264	Connection name
MQCD_REMOTEUSERIDENTIFIER	DS	CL12	First 12 bytes of user identifier from partner
*			
MQCD_REMOTEPASSWORD	DS	CL12	Password from partner
MQCD_MSGRETRYEXIT	DS	CLn	Channel message retry exit name
MQCD_MSGRETRYUSERDATA	DS	CL32	Channel message retry exit user data
*			
MQCD_MSGRETRYCOUNT	DS	F	Number of times MCA will try to put the message, after the first attempt has failed
*			
MQCD_MSGRETRYINTERVAL	DS	F	Minimum interval in milliseconds after which the open or put operation will be retried
*			
MQCD_HEARTBEATINTERVAL	DS	F	Time in seconds between heartbeat flows
*			
MQCD_BATCHINTERVAL	DS	F	Batch duration
MQCD_NONPERSISTENTMSGSPPEED	DS	F	Speed at which nonpersistent messages are sent
*			
MQCD_STRUCLLENGTH	DS	F	Length of MQCD structure

MQCD_EXITNAMELENGTH	DS	F	Length of exit name
MQCD_EXITDATALENGTH	DS	F	Length of exit user data
MQCD_MSGEXITSDEFINED	DS	F	Number of message exits defined
MQCD_SENDEXITSDEFINED	DS	F	Number of send exits defined
MQCD_RECEIVEEXITSDEFINED	DS	F	Number of receive exits defined
MQCD_MSGEXITPTR	DS	F	Address of first MSGEXIT field
MQCD_MSGUSERDATAPTR	DS	F	Address of first MSGUSERDATA field
*			
MQCD_SENDEXITPTR	DS	F	Address of first SENDEXIT field
MQCD_SENDUSERDATAPTR	DS	F	Address of first SENDUSERDATA field
*			
MQCD_RECEIVEEXITPTR	DS	F	Address of first RECEIVEEXIT field
*			
MQCD_RECEIVEUSERDATAPTR	DS	F	Address of first RECEIVEUSERDATA field
*			
MQCD_CLUSTERPTR	DS	F	Address of a list of cluster names
*			
MQCD_CLUSTERSDEFINED	DS	F	Number of clusters to which the channel belongs
*			
MQCD_NETWORKPRIORITY	DS	F	Network priority
MQCD_LONGMCAUSERIDLENGTH	DS	F	Length of long MCA user identifier
*			
MQCD_LONGREMOTEUSERIDLENGTH	DS	F	Length of long remote user identifier
*			
MQCD_LONGMCAUSERIDPTR	DS	F	Address of long MCA user identifier
*			
MQCD_LONGREMOTEUSERIDPTR	DS	F	Address of long remote user identifier
*			
MQCD_MCASECURITYID	DS	XL40	MCA security identifier
MQCD_REMOTESESECURITYID	DS	XL40	Remote security identifier
MQCD_SSLCIPHERSPEC	DS	CL32	TLS CipherSpec
MQCD_SSLPEERNAMEPTR	DS	F	Address of TLS peer name
MQCD_SSLPEERNAMELENGTH	DS	F	Length of TLS peer name
MQCD_SSLCLIENTAUTH	DS	F	Whether TLS client authentication is required
*			
MQCD_KEEPLIVEINTERVAL	DS	F	Keepalive interval
MQCD_LOCALADDRESS	DS	CL48	Local communications address
MQCD_BATCHHEARTBEAT	DS	F	Batch heartbeat interval
MQCD_HDRCOMPLIST	DS	CL2	Header data compression list
MQCD_MSGCOMPLIST	DS	CL16	Message data compression list
MQCD_CLWLCHANNELRANK	DS	F	Channel rank
MQCD_CLWLCHANNELPRIORITY	DS	F	Channel priority
MQCD_CLWLCHANNELWEIGHT	DS	F	Channel weight
MQCD_CHANNELMONITORING	DS	F	Channel monitoring
MQCD_CHANNELSTATISTICS	DS	F	Channel statistics
MQCD_SHARINGCONVERSATIONS	DS	F	Limit on sharing conversations
*			
MQCD_PROPERTYCONTROL	DS	F	Message property control
*			
MQCD_SHARINGCONVERSATIONS	DS	F	Limit on sharing conversations
MQCD_PROPERTYCONTROL	DS	F	Message property control
MQCD_MAXINSTANCES	DS	F	Limit on SVRCONN chl instances
MQCD_MAXINSTANCESPERCLIENT	DS	F	Limit on SVRCONN chl instances per client
MQCD_CLIENTCHANNELWEIGHT	DS	F	Channel weight
MQCD_CONNECTIONAFFINITY	DS	F	Connection Affinty
MQCD_BATCHDATALIMIT	DS	F	Batch data limit
MQCD_USEDLQ	DS	F	Use dead-letter queue
MQCD_DEFRECONNECT	DS	F	Default client reconnect option
MQCD_CERTLABL	DS	F	Certificate label
MQCD_SPLPROTECTION	DS	F	AMS Security policy protection
MQCD_LENGTH	EQU	*-MQCD	
	ORG	MQCD	
MQCD_AREA	DS	CL(MQCD_LENGTH)	

## Déclaration Visual Basic

Cette déclaration est la déclaration Visual Basic de la structure MQCD.

Dans Visual Basic, la structure MQCD peut être utilisée avec la structure MQCNO sur l'appel MQCONN.

Type MQCD			
ChannelName	As String*20		'Channel definition name'
Version	As Long		'Structure version number'
ChannelType	As Long		'Channel type'
TransportType	As Long		'Transport type'
Desc	As String*64		'Channel description'
QMgrName	As String*48		'Queue manager name'
XmitQName	As String*48		'Transmission queue name'

ShortConnectionName	As String*20	'First 20 bytes of connection name'
MCAName	As String*20	'Reserved'
ModeName	As String*8	'LU 6.2 Mode name'
TpName	As String*64	'LU 6.2 transaction program name'
BatchSize	As Long	'Batch size'
DiscInterval	As Long	'Disconnect interval'
ShortRetryCount	As Long	'Short retry count'
ShortRetryInterval	As Long	'Short retry wait interval'
LongRetryCount	As Long	'Long retry count'
LongRetryInterval	As Long	'Long retry wait interval'
SecurityExit	As String*128	'Channel security exit name'
MsgExit	As String*128	'Channel message exit name'
SendExit	As String*128	'Channel send exit name'
ReceiveExit	As String*128	'Channel receive exit name'
SeqNumberWrap	As Long	'Highest allowable message sequence number'
MaxMsgLength	As Long	'Maximum message length'
PutAuthority	As Long	'Put authority'
DataConversion	As Long	'Data conversion'
SecurityUserData	As String*32	'Channel security exit user data'
MsgUserData	As String*32	'Channel message exit user data'
SendUserData	As String*32	'Channel send exit user data'
ReceiveUserData	As String*32	'Channel receive exit user data'
UserIdentifier	As String*12	'User identifier'
Password	As String*12	'Password'
MCAUserIdentifier	As String*12	'First 12 bytes of MCA user identifier'
MCAType	As Long	'Message channel agent type'
ConnectionName	As String*264	'Connection name'
RemoteUserIdentifier	As String*12	'First 12 bytes of user identifier from partner'
RemotePassword	As String*12	'Password from partner'
MsgRetryExit	As String*128	'Channel message retry exit name'
MsgRetryUserData	As String*32	'Channel message retry exit user data'
MsgRetryCount	As Long	'Number of times MCA will try to put the message, after the first attempt has failed'
MsgRetryInterval	As Long	'Minimum interval in milliseconds after which the open or put operation will be retried'
HeartbeatInterval	As Long	'Time in seconds between heartbeat flows'
BatchInterval	As Long	'Batch duration'
NonPersistentMsgSpeed	As Long	'Speed at which nonpersistent messages are sent'
StrucLength	As Long	'Length of MQCD structure'
ExitNameLength	As Long	'Length of exit name'
ExitDataLength	As Long	'Length of exit user data'
MsgExitsDefined	As Long	'Number of message exits defined'
SendExitsDefined	As Long	'Number of send exits defined'
ReceiveExitsDefined	As Long	'Number of receive exits defined'
MsgExitPtr	As MQPTR	'Address of first MsgExit field'
MsgUserDataPtr	As MQPTR	'Address of first MsgUserData field'
SendExitPtr	As MQPTR	'Address of first SendExit field'
SendUserDataPtr	As MQPTR	'Address of first SendUserData field'
ReceiveExitPtr	As MQPTR	'Address of first ReceiveExit field'
ReceiveUserDataPtr	As MQPTR	'Address of first ReceiveUserData field'
ClusterPtr	As MQPTR	'Address of a list of cluster names'
ClustersDefined	As Long	'Number of clusters to which the channel belongs'
NetworkPriority	As Long	'Network priority'
LongMCAUserIdLength	As Long	'Length of long MCA user identifier'
LongRemoteUserIdLength	As Long	'Length of long remote user identifier'
LongMCAUserIdPtr	As MQPTR	'Address of long MCA user identifier'
LongRemoteUserIdPtr	As MQPTR	'Address of long remote user identifier'
MCASecurityId	As MQBYTE40	'MCA security identifier'
RemoteSecurityId	As MQBYTE40	'Remote security identifier'
SSLCipherSpec	As String*32	'TLS CipherSpec'
SSLPeerNamePtr	As MQPTR	'Address of TLS peer name'
SSLPeerNameLength	As Long	'Length of TLS peer name'

SSLClientAuth	As Long	'Whether TLS client authentication is required'
KeepAliveInterval	As Long	'Keepalive interval'
LocalAddress	As String*48	'Local communications address'
BatchHeartbeat	As Long	'Batch heartbeat interval'
HdrCompList(0 to 1)	As Long2	'Header data compression list'
MsgCompList(0 To 15)	As Long16	'Message data compression list'
CLWLChannelRank	As Long	'Channel Rank'
CLWLChannelPriority	As Long	'Channel priority'
CLWLChannelWeight	As Long	'Channel Weight'
ChannelMonitoring	As Long	'Channel Monitoring control'
ChannelStatistics	As Long	'Channel Statistics'
End Type		

### **Modification des zones MQCD dans un exit de canal**

Un exit de canal peut modifier les zones de la base de données MQCD. Toutefois, ces modifications ne sont généralement pas prises en compte, sauf dans les circonstances indiquées.

Si un programme d'exit de canal modifie une zone de la structure de données MQCD, la nouvelle valeur est généralement ignorée par le processus de canal IBM MQ . Toutefois, la nouvelle valeur reste dans le MQCD et est transmise aux exits restants d'une chaîne d'exit et à toute conversation partageant l'instance de canal.

Si SharingConversations est défini sur FALSE dans la structure MQCXP, des modifications peuvent être apportées à certaines zones en fonction du type de programme d'exit, du type de canal et du code anomalie de l'exit. Le tableau suivant présente les zones qui peuvent être modifiées et affecter le comportement du canal, et dans quelles circonstances. Si un programme d'exit modifie l'une de ces zones dans d'autres circonstances ou dans une zone non répertoriée, la nouvelle valeur est ignorée par le processus de canal. La nouvelle valeur reste dans le MQCD et est transmise à tous les exits restants d'une chaîne d'exit et à toute conversation partageant l'instance de canal.

Tout type de programme d'exit lorsqu'il est appelé pour l'initialisation (MQXR\_INIT) peut modifier la zone ChannelName de tout type de canal, tant que MQCXP SharingConversations est défini sur FALSE. Seul un exit de sécurité peut modifier la zone MCAUserIdentifier , quelle que soit la valeur de MQCXP SharingConversations.

Zone	Code raison de sortie	Type de sortie	Type de canal
ChannelName	MQXR_INIT	Tous	Tous
TransportType	MQXR_INIT	Tous	Tous
XmitQName	MQXR_INIT	Tous	SDR, RCVR
ModeName	MQXR_INIT	Tous	Tous
TpName	MQXR_INIT	Tous	Tous
BatchSize	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
DiscInterval	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR



Tableau 823. Zones pouvant être modifiées et affectant le comportement du canal (suite)

<b>Zone</b>	<b>Code raison de sortie</b>	<b>Type de sortie</b>	<b>Type de canal</b>
ShortRetry	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
Intervalle ShortRetry	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
Nombre de LongRetry	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
Intervalle LongRetry	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
SeqNumberRetour à la ligne	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
Longueur maximale des messages	MQXR_INIT	Tous	Tous
PutAuthority	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
DataConversion	MQXR_INIT	Tous	Tous
MCAUserIdentifier	MQXR_INIT, MQXR_INIT_SEC, MQXR_SEC_MSG, MQXR_SEC_PARMS	Sécurité	RCVR, RQSTR, SVRCONN, CLUSRCVR
ConnectionName	MQXR_INIT	Tous	SDR, SVR, RQSTR, CLNTCONN, CLUSSDR, CLUSRCVR
MsgRetryUserData	MQXR_INIT	Tous	RCVR, RQSTR, CLUSRCVR

Tableau 823. Zones pouvant être modifiées et affectant le comportement du canal (suite)

<b>Zone</b>	<b>Code raison de sortie</b>	<b>Type de sortie</b>	<b>Type de canal</b>
MsgRetry	MQXR_INIT	Tous	RCVR, RQSTR, CLUSRCVR
MsgRetryIntervalle	MQXR_INIT	Tous	RCVR, RQSTR, CLUSRCVR
HeartbeatInterval	MQXR_INIT	Tous	Tous
BatchInterval	MQXR_INIT	Tous	SDR, SVR, CLUSSDR, CLUSRCVR
NonPersistentMsgSpeed	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
MCASecurityId	MQXR_INIT, MQXR_INIT_SEC, MQXR_SEC_MSG, MQXR_SEC_PARMS	Sécurité	SDR, SVR, RCVR, RQSTR, SVRCONN, CLUSSDR, CLUSRCVR
SSLCipherSpec	MQXR_INIT	Tous	Tous
SSLPeerNamePtr	MQXR_INIT	Tous	Tous
SSLPeerNameLongueur	MQXR_INIT	Tous	Tous
SSLClientAuth	MQXR_INIT	Tous	RVR, RCVR, RQSTR, SVRCONN, CLUSRCVR
KeepAliveInterval	MQXR_INIT	Tous	Tous
LocalAddress	MQXR_INIT	Tous	SDR, SVR, RQSTR, CLNTCONN, CLUSSDR, CLUSRCVR
BatchHeartbeat	MQXR_INIT	Tous	SDR, SVR, CLUSSDR, CLUSRCVR
Liste HdrComp	MQXR_INIT	Tous	Tous
Liste MsgComp	MQXR_INIT	Tous	Tous

Tableau 823. Zones pouvant être modifiées et affectant le comportement du canal (suite)

Zone	Code raison de sortie	Type de sortie	Type de canal
ChannelMonitoring	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, SVRCONN, CLUSSDR, CLUSRCVR
ChannelStatistics	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
SharingConversations	MQXR_INIT	Tous	SVRCONN, CLNTCONN
PropertyControl	MQXR_INIT	Tous	SDR, SVR, CLUSSDR, CLUSRCVR

## MQCXP-Paramètre d'exit de canal

La structure MQCXP est transmise à chaque type d'exit appelé par un agent MCA (Message Channel Agent), un canal de connexion client ou un canal de connexion serveur.

Voir MQ\_CHANNEL\_EXIT.

Les zones décrites comme "entrée de l'exit" dans les descriptions qui suivent sont ignorées par le canal lorsque l'exit renvoie le contrôle au canal. Les zones d'entrée que l'exit modifie dans le bloc de paramètres d'exit de canal ne seront pas conservées pour son prochain appel. Les modifications apportées aux zones d'entrée/sortie (par exemple, la zone *ExitUserArea*) sont conservées pour les appels de cette instance de l'exit uniquement. Ces modifications ne peuvent pas être utilisées pour transmettre des données entre des exits différents définis sur le même canal ou entre un même exit défini sur des canaux différents.

### Référence associée

«Zones», à la page 1595

Cette rubrique répertorie toutes les zones de la structure MQCXP et décrit chaque zone.

«Déclaration C», à la page 1607

Cette déclaration est la déclaration C pour la structure MQCXP.

«Déclaration COBOL», à la page 1607

Cette déclaration est la déclaration COBOL pour la structure MQCXP.

«Déclaration RPG (ILE)», à la page 1608

Cette déclaration est la déclaration RPG pour la structure MQCXP.

«Déclaration de l'assembleur System/390», à la page 1609

Cette déclaration est la déclaration d'assembleur System/390 pour la structure MQCXP.

### Zones

Cette rubrique répertorie toutes les zones de la structure MQCXP et décrit chaque zone.

*StrucId* (MQCHAR4)

Cette zone indique l'identificateur de structure.

La valeur doit être:

## **MQCXP\_STRUC\_ID**

Identificateur de la structure des paramètres d'exit de canal.

Pour le langage de programmation C, la constante MQCXP\_STRUC\_ID\_ARRAY est également définie ; cette constante a la même valeur que MQCXP\_STRUC\_ID, mais est un tableau de caractères au lieu d'une chaîne.

Il s'agit d'une zone d'entrée de l'exit.

*Version (MQLONG)*

Cette zone indique le numéro de version de la structure.

La valeur dépend de l'environnement:

### **MQCXP\_VERSION\_1**

Structure des paramètres d'exit de canal Version-1 .

### **MQCXP\_VERSION\_2**

Structure des paramètres d'exit de canal Version-2 .

### **MQCXP\_VERSION\_3**

Structure des paramètres d'exit de canal Version-3 .



La zone contient cette valeur dans les systèmes AIX and Linux non répertoriés ailleurs.

### **MQCXP\_VERSION\_4**

Structure des paramètres d'exit de canal Version-4 .

### **MQCXP\_VERSION\_5**

Structure des paramètres d'exit de canal Version-5 .

### **MQCXP\_VERSION\_6**

Structure des paramètres d'exit de canal Version-6 .

### **MQCXP\_VERSION\_8**

Structure des paramètres d'exit de canal Version-8 .



La zone a cette valeur dans z/OS.

### **MQCXP\_VERSION\_9**

Structure des paramètres d'exit de canal Version-9 .

La zone contient cette valeur dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows
-  z/OS

Les zones qui existent uniquement dans les versions plus récentes de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

### **MQCXP\_CURRENT\_VERSION**

Version actuelle de la structure des paramètres d'exit de canal.

La valeur dépend de l'environnement.

**Remarque :** Lorsqu'une nouvelle version de la structure MQCXP est introduite, la présentation de la partie existante n'est pas modifiée. L'exit doit donc vérifier que le numéro de version est égal ou supérieur à la version la plus basse qui contient les zones que l'exit doit utiliser.

Il s'agit d'une zone d'entrée de l'exit.

*ExitId (MQLONG)*

Cette zone indique le type d'exit appelé et est définie à l'entrée de la routine d'exit.

Les valeurs suivantes sont possibles :

**MQXT\_CHANNEL\_SEC\_EXIT**

Exit de sécurité de canal.

**MQXT\_CHANNEL\_MSG\_EXIT**

Exit de message de canal.

**EXIT MQXT\_CHANNEL\_SEND\_EXIT**

Exit d'émission de canal.

**MQXT\_CHANNEL\_RCV\_EXIT**

Exit de réception de canal.

**MQXT\_CHANNEL\_MSG\_RETRY\_EXIT**

Exit de relance de message de canal.

**MQXT\_CHANNEL\_AUTO\_DEF\_EXIT**

Exit de définition automatique de canal.

Sous z/OS, ce type d'exit est pris en charge uniquement pour les canaux de type MQCHT\_CLUSSDR et MQCHT\_CLUSRCVR.

Il s'agit d'une zone d'entrée de l'exit.

*ExitReason (MQLONG)*

Cette zone indique la raison pour laquelle l'exit est appelé et est définie à l'entrée de la routine d'exit.

Il n'est pas utilisé par l'exit de définition automatique. Les valeurs suivantes sont possibles :

**MQXR\_INIT**

Initialisation de l'exit.

Cette valeur indique que l'exit est appelé pour la première fois. Il permet à l'exit d'acquérir et d'initialiser les ressources dont il a besoin (par exemple, la mémoire).

**MQXR\_TERM**

Arrêt de l'exit.

Cette valeur indique que l'exit est sur le point d'être arrêté. L'exit doit libérer toutes les ressources qu'il a acquises depuis son initialisation (par exemple: mémoire).

**MQXR\_MSG**

Traitement d'un message.

Cette valeur indique que l'exit est appelé pour traiter un message. Cette valeur s'applique uniquement aux exits de message de canal.

**MQXR\_XMIT**

Traiter une transmission.

Cette valeur s'applique uniquement aux exits d'émission et de réception de canal.

**MQXR\_SEC\_MSG**

Message de sécurité reçu.

Cette valeur s'applique uniquement aux exits de sécurité de canal.

**MQXR\_INIT\_SEC**

Initiez l'échange de sécurité.

Cette valeur s'applique uniquement aux exits de sécurité de canal.

L'exit de sécurité du récepteur est toujours appelé avec cette raison immédiatement après avoir été appelé avec MQXR\_INIT, afin de lui donner la possibilité de lancer un échange de

sécurité. S'il refuse l'opportunité (en renvoyant MQXCC\_OK à la place de MQXCC\_SEND\_SEC\_MSG ou MQXCC\_SEND\_AND\_REQUEST\_SEC\_MSG), l'exit de sécurité de l'expéditeur est appelé avec MQXR\_INIT\_SEC.

Si l'exit de sécurité du récepteur lance un échange de sécurité (en renvoyant MQXCC\_SEND\_SEC\_MSG ou MQXCC\_SEND\_AND\_REQUEST\_SEC\_MSG), l'exit de sécurité de l'expéditeur n'est jamais appelé avec MQXR\_INIT\_SEC; il est appelé avec MQXR\_SEC\_MSG pour traiter le message du récepteur. (Dans les deux cas, il est d'abord appelé avec MQXR\_INIT.)

Sauf si l'un des exits de sécurité demande l'arrêt du canal (en définissant *ExitResponse* sur MQXCC\_SUPPRESS\_FUNCTION ou MQXCC\_CLOSE\_CHANNEL), l'échange de sécurité doit se terminer du côté qui a lancé l'échange. Par conséquent, si un exit de sécurité est appelé avec MQXR\_INIT\_SEC et qu'il lance un échange, la prochaine fois que l'exit est appelé, il sera appelé avec MQXR\_SEC\_MSG. Cela se produit, qu'il y ait un message de sécurité à traiter par l'exit ou non. Un message de sécurité s'affiche si le partenaire renvoie MQXCC\_SEND\_SEC\_MSG ou MQXCC\_SEND\_AND\_REQUEST\_SEC\_MSG, mais pas si le partenaire renvoie MQXCC\_OK ou s'il n'y a pas d'exit de sécurité au niveau du partenaire. S'il n'y a pas de message de sécurité à traiter, l'exit de sécurité à l'extrémité initiatrice est à nouveau appelé avec un *DataLength* égal à zéro.

### **MQXR\_RETRY**

Relancez un message.

Cette valeur s'applique uniquement aux exits de relance de message.

### **MQXR\_AUTO\_CLUSSDR**

Définition automatique d'un canal émetteur de cluster.

Cette valeur se produit uniquement pour les exits de définition automatique de canal.

### **MQXR\_AUTO\_RECEIVER**

Définition automatique d'un canal récepteur.

Cette valeur se produit uniquement pour les exits de définition automatique de canal.

### **MQXR\_AUTO\_SVRCONN**

Définition automatique d'un canal de connexion serveur.

Cette valeur se produit uniquement pour les exits de définition automatique de canal.

### **MQXR\_AUTO\_CLUSRCVR**

Définition automatique d'un canal récepteur de cluster.

Cette valeur se produit uniquement pour les exits de définition automatique de canal.

### **MQXR\_SEC\_PARMS**

Paramètres de sécurité

Cette valeur s'applique uniquement aux exits de sécurité et indique qu'une structure MQCSP est transmise à l'exit. Pour plus d'informations, voir [«MQCSP-Paramètres de sécurité»](#), à la page 346

### **Remarque :**

1. Si plusieurs exits sont définis pour un canal, ils sont chacun appelés avec MQXR\_INIT lors de l'initialisation de l'agent MCA. En outre, ils sont chacun appelés avec MQXR\_TERM lorsque l'agent MCA est arrêté.
2. Pour l'exit de définition automatique de canal, *ExitReason* n'est pas défini si *Version* est inférieur à MQCXP\_VERSION\_4. La valeur MQXR\_AUTO\_SVRCONN est implicite dans ce cas.

Il s'agit d'une zone d'entrée de l'exit.

#### *ExitResponse (MQLONG)*

Cette zone indique la réponse de l'exit.

Cette zone est définie par l'exit pour communiquer avec l'agent MCA. Il doit s'agir de l'une des valeurs suivantes:

## **MQXCC\_OK**

La sortie a abouti.

- Pour l'exit de sécurité de canal, cette valeur indique que le transfert de message peut maintenant se poursuivre normalement.
- Pour l'exit de relance de message de canal, cette valeur indique que l'agent MCA doit attendre l'intervalle de temps renvoyé par l'exit dans la zone *MsgRetryInterval* de MQCXP, puis relancer le message.

La zone *ExitResponse2* peut contenir des informations supplémentaires.

## **MQXCC\_SUPPRESS\_FUNCTION**

Supprimer la fonction.

- Pour l'exit de sécurité de canal, cette valeur indique que le canal doit être arrêté.
- Pour l'exit de message de canal, cette valeur indique que le message ne doit pas continuer vers sa destination. A la place, l'agent MCA génère un message de rapport d'exception (s'il a été demandé par l'expéditeur du message d'origine) et place le message contenu dans la mémoire tampon d'origine dans la file d'attente de rebut (si l'expéditeur a spécifié MQRO\_DEAD\_LETTER\_Q) ou le supprime (si l'expéditeur a spécifié MQRO\_DISCARD\_MSG).

Pour les messages persistants, si l'expéditeur a spécifié MQRO\_DEAD\_LETTER\_Q, mais que l'insertion dans la file d'attente de rebut échoue ou qu'il n'y a pas de file d'attente de rebut, le message d'origine est laissé dans la file d'attente de transmission et le message de rapport n'est pas généré. Le message d'origine est également laissé dans la file d'attente de transmission si le message de rapport ne peut pas être généré avec succès.

La zone *Feedback* de la structure MQDLH au début du message dans la file d'attente de rebut indique la raison pour laquelle le message a été inséré dans la file d'attente de rebut ; ce code retour est également utilisé dans le descripteur de message du message de rapport d'exception (s'il a été demandé par l'expéditeur).

- Pour l'exit de relance de message de canal, cette valeur indique que l'agent MCA n'attend pas et ne relance pas le message ; à la place, l'agent MCA poursuit immédiatement son traitement d'échec normal (le message est placé dans la file d'attente des messages non livrés ou supprimé, comme indiqué par l'expéditeur du message).
- Pour l'exit de définition automatique de canal, MQXCC\_OK ou MQXCC\_SUPPRESS\_FUNCTION doit être spécifié. Si aucune de ces valeurs n'est spécifiée, MQXCC\_SUPPRESS\_FUNCTION est utilisée par défaut et la définition automatique est abandonnée.

Cette réponse n'est pas prise en charge pour les exits d'émission et de réception de canal.

## **MQXCC\_SEND\_SEC\_MSG**

Envoyer un message de sécurité.

Cette valeur ne peut être définie que par un exit de sécurité de canal. Il indique que l'exit a fourni un message de sécurité qui doit être transmis au partenaire.

## **MQXCC\_SEND\_AND\_REQUEST\_SEC\_MSG**

Envoyer un message de sécurité nécessitant une réponse.

Cette valeur ne peut être définie que par un exit de sécurité de canal. Il indique

- que l'exit a fourni un message de sécurité qui peut être transmis au partenaire, et
- que l'exit requiert une réponse du partenaire. Si aucune réponse n'est reçue, le canal doit être arrêté, car l'exit n'a pas encore décidé si les communications peuvent se poursuivre.

## **MQXCC\_SUPPRESS\_EXIT**

Supprimer l'exit.

- Cette valeur peut être définie par tous les types d'exit de canal autres qu'un exit de sécurité ou un exit de définition automatique. Il supprime tout autre appel de cet exit (comme si son nom était vide dans la définition de canal), jusqu'à l'arrêt du canal, lorsque l'exit est de nouveau appelé avec un *ExitReason* de MQXR\_TERM.

- Si un exit de relance de message renvoie cette valeur, les nouvelles tentatives de messages pour les messages suivants sont contrôlées par les attributs de canal *MsgRetryCount* et *MsgRetryInterval* de manière normale. Pour le message en cours, l'agent MCA effectue le nombre de relances en attente, à des intervalles définis par l'attribut de canal *MsgRetryInterval*, mais uniquement si le code raison est un code raison que l'agent MCA relancera normalement (voir la zone *MsgRetryCount* décrite dans «MQCD-Définition de canal», à la page 1553). Le nombre de nouvelles tentatives en attente correspond à la valeur de l'attribut **MsgRetryCount**, moins le nombre de fois où l'exit a renvoyé MQXCC\_OK pour le message en cours ; si ce nombre est négatif, aucune nouvelle tentative n'est effectuée par l'agent MCA pour le message en cours.

### **MQXCC\_CANAL\_FERMÉ**

Fermez le canal.

Cette valeur peut être définie par n'importe quel type d'exit de canal, à l'exception d'un exit de définition automatique.

Si le partage de conversations n'est pas activé, cette valeur ferme le canal.

Si le partage de conversations est activé, cette valeur met fin à la conversation. Si cette conversation est la seule conversation sur le canal, le canal se ferme également.

Cette zone est une zone d'entrée-sortie de l'exit.

#### *ExitResponse2 (MQLONG)*

Cette zone indique la réponse secondaire de l'exit.

Cette zone est définie sur zéro à l'entrée de la routine d'exit. Il peut être défini par l'exit pour fournir des informations supplémentaires aux fonctions de canal IBM MQ. Il n'est pas utilisé par l'exit de définition automatique.

L'exit peut définir une ou plusieurs des valeurs suivantes. Si plusieurs valeurs sont requises, elles sont ajoutées. Les combinaisons qui ne sont pas valides sont notées ; d'autres combinaisons sont autorisées.

### **MQXR2\_PUT\_WITH\_DEF\_ACTION**

Insertion avec l'action par défaut.

Cette valeur est définie par l'exit de message de canal du récepteur. Il indique que le message doit être inséré avec l'action par défaut de l'agent MCA, c'est-à-dire soit l'ID utilisateur par défaut de l'agent MCA, soit le contexte *UserIdentifier* dans le MQMD (descripteur de message) du message.

La valeur est zéro, ce qui correspond à la valeur initiale définie lors de l'appel de l'exit. La constante est fournie à des fins de documentation.

### **MQXR2\_PUT\_WITH\_DEF\_USERID**

Insertion avec l'ID utilisateur par défaut.

Cette valeur ne peut être définie que par l'exit de message de canal du récepteur. Il indique que le message doit être inséré avec l'ID utilisateur par défaut de l'agent MCA.

### **MQXR2\_PUT\_WITH\_MSG\_USERID**

Insertion avec l'ID utilisateur du message.

Cette valeur ne peut être définie que par l'exit de message de canal du récepteur. Il indique que le message doit être inséré avec le contexte *UserIdentifier* dans le MQMD (descripteur de message) du message (il peut avoir été modifié par l'exit).

Un seul des éléments MQXR2\_PUT\_WITH\_DEF\_ACTION, MQXR2\_PUT\_WITH\_DEF\_USERID et MQXR2\_PUT\_WITH\_MSG\_USERID doit être défini.

### **MQXR2\_USE\_AGENT\_BUFFER**

Utilisez la mémoire tampon de l'agent.

Cette valeur indique que les données à transmettre se trouvent dans *AgentBuffer* et non dans *ExitBufferAddr*.



La valeur est zéro, ce qui correspond à la valeur initiale définie lors de l'appel de l'exit. La constante est fournie à des fins de documentation.

### **MQXR2\_USE\_EXIT\_BUFFER**

Utilisez la mémoire tampon d'exit.

Cette valeur indique que les données à transmettre se trouvent dans *ExitBufferAddress* non dans *AgentBuffer*.

Un seul des paramètres MQXR2\_USE\_AGENT\_BUFFER et MQXR2\_USE\_EXIT\_BUFFER doit être défini.

### **MQXR2\_DEFAULT\_CONTINUATION**

Continuation par défaut.

La suite de l'exit suivant dans la chaîne dépend de la réponse du dernier exit appelé:

- Si MQXCC\_SUPPRESS\_FUNCTION ou MQXCC\_CLOSE\_CHANNEL sont renvoyés, aucun autre exit de la chaîne n'est appelé.
- Sinon, l'exit suivant de la chaîne est appelé.

### **MQXR2\_CONTINUE\_CHAIN**

Passez à l'exit suivant.

### **MQXR2\_SUPPRESS\_CHAIN**

Ignorer les exits restants dans la chaîne.

Il s'agit d'une zone d'entrée-sortie pour l'exit.

*Commentaires en retour (MQLONG)*

Cette zone indique le code retour.

Cette zone est définie sur MQFB\_NONE à l'entrée de la routine d'exit.

Si un exit de message de canal associe la zone *ExitResponse* à la valeur MQXCC\_SUPPRESS\_FUNCTION, la zone *Feedback* indique le code retour qui identifie la raison pour laquelle le message a été placé dans la file d'attente de rebut (message non distribué) et est également utilisée pour envoyer un rapport d'exception, le cas échéant. Dans ce cas, si la zone *Feedback* est MQFB\_NONE, le code retour suivant est utilisé:

### **MQFB\_STOPPED\_BY\_MSG\_EXIT**

Message arrêté par l'exit de message de canal.

La valeur renvoyée dans cette zone par les exits de sécurité de canal, d'envoi, de réception et de relance de message n'est pas utilisée par l'agent MCA.

La valeur renvoyée dans cette zone par les exits de définition automatique n'est pas utilisée si *ExitResponse* est MQXCC\_OK, mais elle est utilisée pour le paramètre *AuxErrorDataInt1* dans le message d'événement.

Il s'agit d'une zone d'entrée-sortie de l'exit.

*Longueur MaxSegment(MQLONG)*

Cette zone indique la longueur maximale en octets pouvant être envoyée en une seule transmission.

Il n'est pas utilisé par l'exit de définition automatique. Il est intéressant pour un exit d'émission de canal, car cet exit doit s'assurer qu'il n'augmente pas la taille d'un segment de transmission à une valeur supérieure à *MaxSegmentLength*. La longueur inclut les 8 octets initiaux que l'exit ne doit pas modifier. La valeur est négociée entre les fonctions de canal IBM MQ lorsque le canal est lancé. Pour plus d'informations sur les longueurs de segment, voir [Ecriture de programmes d'exit de canal](#).

La valeur de cette zone n'est pas significative si *ExitReason* est MQXR\_INIT.

Il s'agit d'une zone d'entrée de l'exit.

*Zone ExitUser(MQBYTE16)*

Cette zone indique la zone utilisateur de l'exit-une zone disponible pour l'exit.

Il est initialisé à zéro binaire avant le premier appel de l'exit (qui a un *ExitReason* défini sur MQXR\_INIT), puis toutes les modifications apportées à cette zone par l'exit sont conservées entre les appels de l'exit.

La valeur suivante est définie:

#### **MQXUA\_NONE**

Aucune information utilisateur.

La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQXUA\_NONE\_ARRAY est également définie ; cette constante a la même valeur que MQXUA\_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La longueur de cette zone est indiquée par MQ\_EXIT\_USER\_AREA\_LENGTH. Il s'agit d'une zone d'entrée-sortie pour l'exit.

#### *ExitData (MQCHAR32)*

Cette zone indique les données d'exit.

Cette zone est définie à l'entrée de la routine d'exit sur les informations que les fonctions de canal IBM MQ ont extrait de la définition de canal. Si aucune information de ce type n'est disponible, cette zone est vide.

La longueur de cette zone est donnée par MQ\_EXIT\_DATA\_LENGTH.

Il s'agit d'une zone d'entrée de l'exit.

Les zones suivantes de cette structure ne sont pas présentes si *Version* est inférieur à MQCXP\_VERSION\_2.

#### *MsgRetry(MQLONG)*

Cette zone indique le nombre de fois où le message a été relancé.

La première fois que l'exit est appelé pour un message particulier, cette zone a la valeur zéro (aucune nouvelle tentative n'a encore été effectuée). Lors de chaque appel ultérieur de l'exit pour ce message, la valeur est incrémentée d'un par l'agent MCA.

Il s'agit d'une zone d'entrée de l'exit. La valeur de cette zone n'est pas significative si *ExitReason* est MQXR\_INIT. La zone n'est pas présente si *Version* est inférieur à MQCXP\_VERSION\_2.

#### *Intervalle MsgRetry(MQLONG)*

Cette zone indique l'intervalle minimal en millisecondes après lequel l'opération d'insertion est relancée.

La première fois que l'exit est appelé pour un message particulier, cette zone contient la valeur de l'attribut de canal *MsgRetryInterval*. L'exit peut laisser la valeur inchangée ou la modifier pour spécifier un intervalle de temps différent en millisecondes. Si l'exit renvoie MQXCC\_OK dans *ExitResponse*, l'agent MCA attend au moins cet intervalle de temps avant de relancer l'opération MQOPEN ou MQPUT. L'intervalle de temps spécifié doit être supérieur ou égal à zéro.

La seconde fois et les suivantes où l'exit est appelé pour ce message, cette zone contient la valeur renvoyée par l'appel précédent de l'exit.

Si la valeur renvoyée dans la zone *MsgRetryInterval* est inférieure à zéro ou supérieure à 999 999 999 et que *ExitResponse* est MQXCC\_OK, l'agent MCA ignore la zone *MsgRetryInterval* dans MQCXP et attend à la place l'intervalle spécifié par l'attribut de canal *MsgRetryInterval*.

Il s'agit d'une zone d'entrée-sortie pour l'exit. La valeur de cette zone n'est pas significative si *ExitReason* est MQXR\_INIT. La zone n'est pas présente si *Version* est inférieur à MQCXP\_VERSION\_2.

#### *MsgRetryReason (MQLONG)*

Cette zone indique le code raison de la tentative précédente d'insertion du message.

Cette zone est le code raison de la tentative précédente d'insertion du message ; il s'agit de l'une des valeurs MQRC\_\*.

Il s'agit d'une zone d'entrée de l'exit. La valeur de cette zone n'est pas significative si *ExitReason* est MQXR\_INIT. La zone n'est pas présente si *Version* est inférieur à MQCXP\_VERSION\_2.

Les zones suivantes de cette structure ne sont pas présentes si *Version* est inférieur à MQCXP\_VERSION\_3.

#### *HeaderLength (MQLONG)*

Cette zone indique la longueur des informations d'en-tête.

Cette zone est pertinente uniquement pour un exit de message et un exit de relance de message. La valeur correspond à la longueur des structures d'en-tête de routage au début des données de message ; il s'agit de la structure MQXQH, de MQMDE (en-tête d'extension de description de message) et (pour un message de liste de distribution) de la structure MQDH et des tableaux d'enregistrements MQOR et MQPMR qui suivent la structure MQXQH.

L'exit de message peut examiner ces informations d'en-tête et les modifier si nécessaire, mais les données renvoyées par l'exit doivent toujours être au format correct. L'exit ne doit pas, par exemple, chiffrer ou compresser les données d'en-tête à l'extrémité émettrice, même si l'exit de message à l'extrémité réceptrice effectue des modifications de compensation.

Si l'exit de message modifie les informations d'en-tête de manière à modifier sa longueur (par exemple, en ajoutant une autre destination à un message de liste de distribution), il doit modifier la valeur de *HeaderLength* en conséquence avant de renvoyer le message.

Il s'agit d'une zone d'entrée-sortie pour l'exit. La valeur de cette zone n'est pas significative si *ExitReason* est MQXR\_INIT. La zone n'est pas présente si *Version* est inférieur à MQCXP\_VERSION\_3.

#### *PartnerName (MQCHAR48)*

Cette zone indique le nom du partenaire.

Nom du partenaire, comme suit:

- Pour les canaux SVRCONN, il s'agit de l'ID utilisateur connecté sur le client.
- Pour tous les autres types de canal, il s'agit du nom du gestionnaire de files d'attente du partenaire.

Lorsque l'exit est initialisé, cette zone est vide car le gestionnaire de files d'attente ne connaît pas le nom du partenaire tant que la négociation initiale n'a pas eu lieu.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP\_VERSION\_3.

#### *FAPLevel (MQLONG)*

Niveau des formats et protocoles négociés.

Il s'agit d'une zone d'entrée de l'exit. Les modifications apportées à cette zone ne doivent être effectuées que sur instruction du service IBM . La zone n'est pas présente si *Version* est inférieur à MQCXP\_VERSION\_3.

#### *CapabilityFlags (MQLONG)*

Vous pouvez définir l'indicateur de capacité sur MQCF\_NONE ou sur MQCF\_DIST\_LISTS.

Vous pouvez définir l'un des indicateurs de capacité suivants:

#### **MQCF\_AUCUN**

Aucun indicateur.

#### **LIST\_LISTES MQCF\_DIST**

Listes de distribution prises en charge.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP\_VERSION\_3.

#### *ExitNumber (MQLONG)*

Cette zone indique le nombre ordinal de l'exit.

Nombre ordinal de l'exit, dans le type défini dans *ExitId*. Par exemple, si l'exit appelé est le troisième exit de message défini, cette zone contient la valeur 3. Si le type d'exit est un type pour lequel une liste d'exits ne peut pas être définie (par exemple, un exit de sécurité), cette zone a la valeur 1.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP\_VERSION\_3.

Les zones suivantes de cette structure ne sont pas présentes si *Version* est inférieur à MQCXP\_VERSION\_5.

#### *ExitSpace (MQLONG)*

Cette zone indique le nombre d'octets dans la mémoire tampon de transmission réservée à l'exit à utiliser.

Cette zone est pertinente uniquement pour un exit d'émission. Il indique la quantité d'espace en octets que le canal IBM MQ réserve dans la mémoire tampon de transmission pour que l'exit puisse l'utiliser. Ce champ permet à la sortie d'ajouter à la mémoire tampon de transmission une petite quantité de données (typiquement ne dépassant pas quelques centaines d'octets) à utiliser par une sortie de réception complémentaire à l'autre extrémité. Les données ajoutées par l'exit d'émission doivent être supprimées par l'exit de réception.

La valeur est toujours zéro sous z/OS.

**Remarque :** Cette fonction ne doit pas être utilisée pour envoyer de grandes quantités de données, car elle risque de dégrader les performances, voire d'empêcher le fonctionnement du canal.

En définissant *ExitSpace*, l'exit est garanti qu'il y a toujours au moins le nombre d'octets disponibles dans la mémoire tampon de transmission à utiliser par l'exit. Toutefois, l'exit peut utiliser moins que la quantité réservée ou plus que la quantité réservée s'il y a de l'espace disponible dans la mémoire tampon de transmission. L'espace de sortie dans la mémoire tampon est fourni après les données existantes.

*ExitSpace* peut être défini par l'exit uniquement lorsque *ExitReason* a la valeur MQXR\_INIT; dans tous les autres cas, la valeur renvoyée par l'exit est ignorée. En entrée de l'exit, *ExitSpace* est égal à zéro pour l'appel MQXR\_INIT et correspond à la valeur renvoyée par l'appel MQXR\_INIT dans les autres cas.

Si la valeur renvoyée par l'appel MQXR\_INIT est négative ou qu'il y a moins de 1024 octets disponibles dans la mémoire tampon de transmission pour les données de message après la réservation de l'espace d'exit demandé pour tous les exits d'émission de la chaîne, l'agent MCA génère un message d'erreur et ferme le canal. De même, si, lors du transfert de données, les exits de la chaîne d'exit d'émission allouent plus d'espace utilisateur qu'ils n'en réservent de sorte qu'il reste moins de 1024 octets dans la mémoire tampon de transmission pour les données de message, l'agent MCA génère un message d'erreur et ferme le canal. La limite de 1024 permet aux flux de contrôle et d'administration du canal d'être traités par la chaîne d'exits d'émission, sans qu'il soit nécessaire de segmenter les flux.

Il s'agit d'une zone d'entrée-sortie pour l'exit si *ExitReason* est MQXR\_INIT, et d'une zone d'entrée dans tous les autres cas. La zone n'est pas présente si *Version* est inférieur à MQCXP\_VERSION\_5.

#### *ID SSLCertUser(MQCHAR12)*

Cette zone indique l' *UserId* associé au certificat distant.

Elle est vide sur toutes les plateformes, à l'exception de z/OS

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP\_VERSION\_6.

#### *SSLRemCertIssNameLongueur (MQLONG)*

Cette zone indique la longueur en octets du nom distinctif complet de l'émetteur du certificat distant désigné par *SSLCertRemoteIssuerNamePtr*.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP\_VERSION\_6. La valeur est zéro s'il ne s'agit pas d'un canal TLS.

#### *SSLRemCertIssNamePtr (PMQVOID)*

Cette zone indique l'adresse du nom distinctif complet de l'émetteur du certificat distant.

Sa valeur est le pointeur null s'il ne s'agit pas d'un canal TLS.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP\_VERSION\_6.

**Remarque :** Le comportement des exits de sécurité de canal lors de la détermination du nom distinctif du sujet et du nom distinctif de l'émetteur est modifié à partir de IBM WebSphere MQ 7.1. Pour plus d'informations, voir [Programmes d'exit de sécurité de canal](#).

#### *SecurityParms (PMQCSP)*

Cette zone indique l'adresse de la structure MQCSP utilisée pour spécifier les données d'authentification.

La valeur initiale de cette zone est le pointeur null.

Il s'agit d'une zone d'entrée-sortie pour l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP\_VERSION\_6.

La valeur renvoyée par l'exit dans cette zone doit être utilisable par IBM MQ jusqu'à MQXR\_TERM.

#### *Compression CurHdr(MQLONG)*

Cette zone indique la technique actuellement utilisée pour compresser les données d'en-tête.

Il est défini sur l'une des valeurs suivantes:

##### **MQCOMPRESS\_NONE**

Aucune compression de données d'en-tête n'est effectuée.

##### **MQCOMPRESS\_SYSTEM**

La compression de données d'en-tête est effectuée.

La valeur peut être modifiée par l'exit de message d'un canal émetteur en une des valeurs prises en charge négociées accessibles à partir de la zone de liste HdrCompde MQCD. Cela permet la technique utilisée pour compresser les données d'en-tête à choisir pour chaque message en fonction du contenu du message. La valeur modifiée est utilisée pour le message en cours uniquement. Le canal se termine si l'attribut est modifié avec une valeur non prise en charge. La valeur est ignorée si elle est modifiée en dehors de l'exit de message d'un canal émetteur.

Il s'agit d'une zone d'entrée-sortie pour l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP\_VERSION\_6.

#### *Compression CurMsg(MQLONG)*

Cette zone indique la technique actuellement utilisée pour compresser les données de message.

Il est défini sur l'une des valeurs suivantes:

##### **MQCOMPRESS\_NONE**

Aucune compression de données d'en-tête n'est effectuée.

##### **MQCOMPRESS\_RLE**

La compression de données de message est effectuée à l'aide de l'algorithme RLE.

##### **MQCOMPRESS\_ZLIBFAST**

La compression de données de message est effectuée à l'aide de la technique de compression zlib. Il est préférable d'utiliser une durée de compression rapide.

##### **MQCOMPRESS\_ZLIBHIGH**

La compression de données de message est effectuée à l'aide de la technique de compression zlib. Il est préférable d'utiliser une compression de haut niveau.

La valeur peut être modifiée par l'exit de message d'un canal émetteur sur l'une des valeurs prises en charge négociées accessibles à partir de la zone de liste MsgCompdu MQCD. Cela permet la technique utilisée pour compresser les données de message à décider pour chaque message en fonction du contenu du message. La valeur modifiée est utilisée pour le message en cours uniquement. Le canal se termine si l'attribut est modifié avec une valeur non prise en charge. La valeur est ignorée si elle est modifiée en dehors de l'exit de message d'un canal émetteur.

Il s'agit d'une zone d'entrée-sortie pour l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP\_VERSION\_6.

#### *Hconn (MQHCONN)*

Cette zone indique le descripteur de connexion utilisé par l'exit s'il doit effectuer des appels MQI dans l'exit.

Cette zone n'est pas pertinente pour les exits exécutés sur des canaux de connexion client, où elle contient la valeur MQHC\_UNUSABLE\_HCONN (-1).

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP\_VERSION\_7.

#### *SharingConversations (MQBOOL)*

Cette zone indique si la conversation est la seule qui peut être en cours d'exécution sur cette instance de canal ou si plusieurs conversations peuvent être en cours d'exécution sur cette instance de canal.

Elle indique également si le programme d'exit est soumis au risque que le MQCD soit modifié par un autre programme d'exit exécuté en même temps.

Cette zone concerne uniquement les programmes d'exit exécutés sur des canaux de connexion client ou serveur.

Il est défini sur l'une des valeurs suivantes:

#### **FALSE**

L'instance d'exit est la seule instance d'exit qui peut actuellement être en cours d'exécution sur cette instance de canal. Cela permet à l'exit de mettre à jour en toute sécurité les zones MQCD sans conflit avec d'autres exits s'exécutant sur d'autres instances de canal. Le fait que les modifications apportées aux zones MQCD soient traitées par le canal est défini par la table des zones MQCD dans [«Modification des zones MQCD dans un exit de canal»](#), à la page 1592.

#### **exit utilisateur associé à une tâche**

L'instance d'exit n'est pas la seule instance d'exit qui peut être en cours d'exécution sur cette instance de canal. Les modifications apportées à MQCD ne sont pas traitées par le canal, à l'exception des modifications répertoriées dans la table des zones MQCD dans [«Modification des zones MQCD dans un exit de canal»](#), à la page 1592 pour les raisons d'exit autres que MQXR\_INIT. Si cet exit met à jour les zones MQCD, vérifiez qu'il n'y a pas de conflit d'autres exits qui s'exécutent simultanément sur d'autres conversations en fournissant une sérialisation parmi les exits qui s'exécutent sur cette instance de canal.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP\_VERSION\_7.

#### *MCAUserSource (MQLONG)*

Cette zone indique la source de l'ID utilisateur MCA fourni.

Il peut contenir l'une des valeurs suivantes:

#### **MQUSRC\_MAP**

L'ID utilisateur est indiqué dans l'attribut MCAUSER.

#### **MQUSRC\_CHANNEL**

L'ID utilisateur est transmis à partir du partenaire entrant ou spécifié dans la zone MCAUSER définie dans l'objet canal.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si la version est inférieure à MQCXP\_VERSION\_8.

#### *Points pEntry(PMQIEP)*

Cette zone indique l'adresse du point d'entrée d'interface pour l'appel MQI ou DCI.

La zone n'est pas présente si *Version* est inférieure à MQCXP\_VERSION\_8.

#### *RemoteProduct (MQCHAR4)*

Cette zone indique le nom du produit distant.

Cette zone identifie le produit distant du client, par exemple, C ou Java, tel qu'affiché dans la zone **RPRODUCT** de [DISPLAY CHSATUS](#).

La zone n'est pas présente si *Version* est inférieure à MQCXP\_VERSION\_9.

#### *RemoteVersion (MQCHAR8)*

Cette zone indique le nom de la version distante.

Cette zone identifie la version des bibliothèques client, comme indiqué dans la zone **RVERSION** de DISPLAY CHSTATUS.

La zone n'est pas présente si *Version* est inférieure à MQCXP\_VERSION\_9.

### **Déclaration C**

Cette déclaration est la déclaration C pour la structure MQCXP.

```
typedef struct tagMQCXP MQCXP;
struct tagMQCXP {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    ExitId;           /* Type of exit */
    MQLONG    ExitReason;       /* Reason for invoking exit */
    MQLONG    ExitResponse;     /* Response from exit */
    MQLONG    ExitResponse2;    /* Secondary response from exit */
    MQLONG    Feedback;         /* Feedback code */
    MQLONG    MaxSegmentLength; /* Maximum segment length */
    MQBYTE16  ExitUserArea;     /* Exit user area */
    MQCHAR32  ExitData;         /* Exit data */
    MQLONG    MsgRetryCount;    /* Number of times the message has been
    MQLONG    MsgRetryInterval; /* Minimum interval in milliseconds after
    MQLONG    MsgRetryReason;   /* Reason code from previous attempt to
    MQLONG    HeaderLength;     /* Length of header information */
    MQCHAR48  PartnerName;     /* Partner Name */
    MQLONG    FAPLevel;        /* Negotiated Formats and Protocols
    MQLONG    CapabilityFlags;  /* Capability flags */
    MQLONG    ExitNumber;      /* Exit number */
    /* Ver:3 */
    /* Ver:4 */
    MQLONG    ExitSpace;        /* Number of bytes in transmission buffer
    /* Ver:5 */
    MQCHAR12  SSLCertUserid;    /* User identifier associated
    MQLONG    SSLRemCertIssNameLength; /* Length of
    MQPTR     SSLRemCertIssNamePtr; /* Address of
    PMQVOID   SecurityParms;    /* Security parameters */
    MQLONG    CurHdrCompression; /* Header data compression
    MQLONG    CurMsgCompression; /* Message data compression
    /* Ver:6 */
    MQHCONN   Hconn;           /* Connection handle */
    MQBOOL    SharingConversations; /* Multiple conversations
    /* Ver:7 */
    MQLONG    MCAUserSource;    /* Source of the provided MCA user ID */
    PMQIEP    pEntryPoints;     /* Address of the MQIEP structure */
    /* Ver:8 */
    MQCHAR4   RemoteProduct;    /* The identifier for the remote product */
    MQCHAR8   RemoteVersion;    /* The version of the remote product */
    /* Ver:9 */
};
```

### **Déclaration COBOL**

Cette déclaration est la déclaration COBOL pour la structure MQCXP.

```
** MQCXP structure
```

```

10 MQCXP.
** Structure identifier
15 MQCXP-STRUCID PIC X(4).
** Structure version number
15 MQCXP-VERSION PIC S9(9) BINARY.
** Type of exit
15 MQCXP-EXITID PIC S9(9) BINARY.
** Reason for invoking exit
15 MQCXP-EXITREASON PIC S9(9) BINARY.
** Response from exit
15 MQCXP-EXITRESPONSE PIC S9(9) BINARY.
** Secondary response from exit
15 MQCXP-EXITRESPONSE2 PIC S9(9) BINARY.
** Feedback code
15 MQCXP-FEEDBACK PIC S9(9) BINARY.
** Maximum segment length
15 MQCXP-MAXSEGMENTLENGTH PIC S9(9) BINARY.
** Exit user area
15 MQCXP-EXITUSERAREA PIC X(16).
** Exit data
15 MQCXP-EXITDATA PIC X(32).
** Number of times the message has been retried
15 MQCXP-MSGRETRYCOUNT PIC S9(9) BINARY.
** Minimum interval in milliseconds after which the put operation
** should be retried
15 MQCXP-MSGRETRYINTERVAL PIC S9(9) BINARY.
** Reason code from previous attempt to put the message
15 MQCXP-MSGRETRYREASON PIC S9(9) BINARY.
** Length of header information
15 MQCXP-HEADERLENGTH PIC S9(9) BINARY.
** Partner Name
15 MQCXP-PARTNERNAME PIC X(48).
** Negotiated Formats and Protocols level
15 MQCXP-FAPLEVEL PIC S9(9) BINARY.
** Capability flags
15 MQCXP-CAPABILITYFLAGS PIC S9(9) BINARY.
** Exit number
15 MQCXP-EXITNUMBER PIC S9(9) BINARY.
** Number of bytes in transmission buffer reserved for exit to use
15 MQCXP-EXITSPACE PIC S9(9) BINARY.
** User Id associated with remote certificate
15 MQCXP-SSLCERTUSERID PIC X(12).
** Length of distinguished name of issuer of remote TLS
** certificate
15 MQCXP-SSLREMCERTISSNAMELENGTH PIC S9(9) BINARY.
** Address of distinguished name of issuer of remote TLS
** certificate
15 MQCXP-SSLREMCERTISSNAMEPTR POINTER.
** Security parameters
15 MQCXP-SECURITYPARMS PIC S9(18) BINARY.
** Header data compression used for current message
15 MQCXP-CURHDRCOMPRESSION PIC S9(9) BINARY.
** Message data compression used for current message
15 MQCXP-CURMSGCOMPRESSION PIC S9(9) BINARY.
** Connection handle
15 MQCXP-HCONN PIC S9(9) BINARY.
** Multiple conversations possible on channel instance?
15 MQCXP-SHARINGCONVERSATIONS PIC S9(9) BINARY.
** Source of the provided MCA user ID
15 MQCXP-MCAUSERSOURCE PIC S9(9) BINARY.
** Identifier of the remote product
15 MQCXP-RPRODUCT PIC X(4).
** Identifier of the remote version
15 MQCXP-RVERSION PIC X(8).

```

## Déclaration RPG (ILE)

Cette déclaration est la déclaration RPG pour la structure MQCXP.

```

D*..1.....2.....3.....4.....5.....6.....7..
D* MQCXP Structure
D*
D* Structure identifier
D CXSID 1 4
D* Structure version number
D CXVER 5 8I 0
D* Type of exit
D CXXID 9 12I 0
D* Reason for invoking exit

```



```

D CXREA          13      16I 0
D* Response from exit
D CXRES          17      20I 0
D* Secondary response from exit
D CXRE2          21      24I 0
D* Feedback code
D CXFB           25      28I 0
D* Maximum segment length
D CXMSL          29      32I 0
D* Exit user area
D CXUA           33      48
D* Exit data
D CXDAT          49      80
D* Number of times the message has been retried
D CXMRC          81      84I 0
D* Minimum interval in milliseconds after which the put operation
D* should be retried
D CXMRI          85      88I 0
D* Reason code from previous attempt to put the message
D CXMRR          89      92I 0
D* Length of header information
D CXHDL          93      96I 0
D* Partner Name
D CXPNM          97      144
D* Negotiated Formats and Protocols level
D CXFAP         145      148I 0
D* Capability flags
D CXCAP         149      152I 0
D* Exit number
D CXEXN         153      156I 0
D* Number of bytes in transmission buffer reserved for exit to use
D CXHDL         157      160I 0
D* User identifier associated with remote TLS certificate
D CXSSLCU       161      172
D* Length of distinguished name of issuer of remote TLS certificate
D CXSRCINL     173      176I 0
D* Address of distinguished name of issuer of remote TLS certificate
D CXSRCINP     177      192*
D* Security parameters
D CXSECP       193      208*
D* Header data compression used for current message
D CXCHC       209      212I 0
D* Message data compression used for current message
D CXCMC       213      216I 0
D* Connection handle
D CXHCONN     217      220I 0
D* Multiple conversations possible on channel instance?
D CXSHARECONV 221      224I 0
D* Source of the provided MCA user ID
D MCAUSERSOURCE 225      228I 0
D* Identifier of the remote product
D CXRPRO      229      232I 0
D* Identifier of the remote version
D CXRVER      233      240I 0

```

## ***Déclaration de l'assembleur System/390***

Cette déclaration est la déclaration d'assembleur System/390 pour la structure MQCXP.

```

MQCXP           DSECT
MQCXP_STRUCID  DS   CL4   Structure identifier
MQCXP_VERSION  DS   F     Structure version number
MQCXP_EXITID   DS   F     Type of exit
MQCXP_EXITREASON DS   F   Reason for invoking exit
MQCXP_EXITRESPONSE DS   F   Response from exit
MQCXP_EXITRESPONSE2 DS   F   Secondary response from exit
MQCXP_FEEDBACK DS   F     Feedback code
MQCXP_MAXSEGMENTLENGTH DS   F   Maximum segment length
MQCXP_EXITUSERAREA DS   XL16 Exit user area
MQCXP_EXITDATA DS   CL32  Exit data
MQCXP_MSGRETRYCOUNT DS   F   Number of times the message has been
*                               retried
MQCXP_MSGRETRYINTERVAL DS   F   Minimum interval in milliseconds
*                               after which the put operation should
*                               be retried
MQCXP_MSGRETRYREASON DS   F   Reason code from previous attempt to
*                               put the message
MQCXP_HEADERLENGTH DS   F   Length of header information
MQCXP_PARTNERNAME DS   CL48  Partner Name

```

MQCXP_FAPLEVEL	DS	F	Negotiated Formats and Protocols level
* MQCXP_CAPABILITYFLAGS	DS	F	Capability flags
MQCXP_EXITNUMBER	DS	F	Exit number
MQCXP_EXITSPLACE	DS	F	Number of bytes in transmission buffer reserved for exit to use
* MQCXP_SSLCERTUSERID	DS	CL12	User identifier associated with remote TLS certificate
* MQCXP_SSLREMCERTISSNAMELENGTH	DS	F	Length of distinguished name of issuer of remote TLS certificate
* MQCXP_SSLREMCERTISSNAMEPTR	DS	F	Address of distinguished name of issuer of remote TLS certificate
* MQCXP_SECURITYPARMS	DS	F	Address of security parameters
MQCXP_CURHDRCOMPRESSION	DS	F	Header data compression used for current message
* MQCXP_CURMSGCOMPRESSION	DS	F	Message data compression used for current message
* MQCXP_HCONN	DS	F	Connection handle
MQCXP_SHARINGCONVERSATIONS	DS	F	Multiple conversations possible on channel inst?
* MQCXP_MCAUSERSOURCE	DS	F	Source of the provided MCA user ID
MQCXP_RPRODUCT	DS	CL4	Identifier of the remote product
MQCXP_RVERSION	DS	CL8	Identifier of the remote version
MQCXP_LENGTH	EQU	*-MQCXP	
	ORG	MQCXP	
MQCXP_AREA	DS	CL(MQCXP_LENGTH)	

## MQXWD-Descripteur d'attente d'exit

La structure MQXWD est un paramètre d'entrée-sortie dans l'appel MQXWAIT.

Cette structure est prise en charge uniquement sous z/OS.

### Référence associée

«Zones», à la page 1610

Cette rubrique répertorie toutes les zones de la structure MQXWD et décrit chaque zone.

«Déclaration C», à la page 1611

Cette déclaration est la déclaration C pour la structure MQXWD.

«Déclaration de l'assembleur System/390», à la page 1611

Cette déclaration est la déclaration d'assembleur System/390 pour la structure MQXWD.

### Zones

Cette rubrique répertorie toutes les zones de la structure MQXWD et décrit chaque zone.

#### StrucId (MQCHAR4)

Cette zone indique l'identificateur de structure.

La valeur doit être:

#### ID\_STRUCD\_MQXWD\_

Identificateur de la structure de descripteur d'attente d'exit.

Pour le langage de programmation C, la constante MQXWD\_STRUC\_ID\_ARRAY est également définie ; cette constante a la même valeur que MQXWD\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La valeur initiale de cette zone est MQXWD\_STRUC\_ID.

#### Version (MQLONG)

Cette zone indique le numéro de version de la structure.

La valeur doit être:

#### MQXWD\_VERSION\_1

Numéro de version de la structure de descripteur d'attente d'exit.

La valeur initiale de cette zone est MQXWD\_VERSION\_1.

### *Reserved1 (MQLONG)*

Cette zone est réservée. Sa valeur doit être zéro.

Il s'agit d'une zone d'entrée.

### *Reserved2 (MQLONG)*

Cette zone est réservée. Sa valeur doit être zéro.

Il s'agit d'une zone d'entrée.

### *Reserved3 (MQLONG)*

Cette zone est réservée. Sa valeur doit être zéro.

Il s'agit d'une zone d'entrée.

### *ECB (MQLONG)*

Cette zone indique le bloc de contrôle d'événement à attendre.

Cette zone indique le bloc de contrôle d'événement (ECB) à attendre. Il doit être défini sur zéro avant que l'appel MQXWAIT ne soit émis ; une fois terminé, il contient le code postal.

Cette zone est une zone d'entrée/sortie.

## **Déclaration C**

Cette déclaration est la déclaration C pour la structure MQXWD.

```
typedef struct tagMQXWD MQXWD;
struct tagMQXWD {
    MQCHAR4  StrucId;      /* Structure identifier */
    MQLONG   Version;     /* Structure version number */
    MQLONG   Reserved1;   /* Reserved */
    MQLONG   Reserved2;   /* Reserved */
    MQLONG   Reserved3;   /* Reserved */
    MQLONG   ECB;        /* Event control block to wait on */
};
```

## **Déclaration de l'assembleur System/390**

Cette déclaration est la déclaration d'assembleur System/390 pour la structure MQXWD.

```
MQXWD          DSECT
MQXWD_STRUCID  DS    CL4  Structure identifier
MQXWD_VERSION  DS    F    Structure version number
MQXWD_RESERVED1 DS    F    Reserved
MQXWD_RESERVED2 DS    F    Reserved
MQXWD_RESERVED3 DS    F    Reserved
MQXWD_ECB      DS    F    Event control block to wait on
*
MQXWD_LENGTH   EQU    *-MQXWD
                ORG    MQXWD
MQXWD_AREA     DS    CL(MQXWD_LENGTH)
```

## **Appel d'exit de charge de travail de cluster et structures de données**

Cette section fournit des informations de référence pour l'exit de charge de travail de cluster et les structures de données. Il s'agit d'informations d'interface de programmation à usage général.


Vous pouvez écrire des exits de charge de travail de cluster dans les langages de programmation suivants:

- C
- Assembleur System/390 ( IBM MQ for z/OS )

L'appel est décrit dans:

- [«MQ\\_CLUSTER\\_WORKLOAD\\_EXIT -Description de l'appel», à la page 1612](#)

Les types de données de structure utilisés par l'exit sont décrits dans:

- «MQXCLWLN -Navigation dans les enregistrements de charge de travail de cluster», à la page 1614
- «MQWXP -Structure des paramètres d'exit de charge de travail de cluster», à la page 1618
- «MQWDR-Structure d'enregistrement de destination de charge de travail de cluster», à la page 1626
- «MQWQR -Structure d'enregistrement de file d'attente de charge de travail de cluster», à la page 1631
- «MQWCR -Structure d'enregistrement de cluster de charge de travail de cluster», à la page 1636
-  Comportement asynchrone des commandes CLUSTER sous z/OS

Dans cette section, les attributs de gestionnaire de files d'attente et les attributs de file d'attente sont affichés en intégralité. Les noms équivalents utilisés dans les commandes MQSC sont indiqués ci-dessous. Pour plus de détails sur les commandes MQSC, voir [Commandes MQSC](#).

*Tableau 824. Attributs du gestionnaire de files d'attente*

Nom complet	Nom utilisé dans MQSC
<i>ClusterWorkloadData</i>	CLWLDATA
<i>ClusterWorkloadExit</i>	CLWLEXIT
<i>ClusterWorkloadLength</i>	CLWLEN

*Tableau 825. Attributs File d'attente*

Nom complet	Nom utilisé dans MQSC
<i>DefBind</i>	DEFBIND
<i>DefPersistence</i>	DEFPSIST
<i>DefPriority</i>	DEFPRTY
<i>InhibitPut</i>	PUT
<i>QDesc</i>	DESCR

### Tâches associées

[Ecriture et compilation des exits de charge de travail de cluster](#)

## MQ\_CLUSTER\_WORKLOAD\_EXIT -Description de l'appel

L'exit de charge de travail de cluster est appelé par le gestionnaire de files d'attente pour acheminer un message vers un gestionnaire de files d'attente disponible.

**Remarque :** Aucun point d'entrée appelé MQ\_CLUSTER\_WORKLOAD\_EXIT n'est fourni par le gestionnaire de files d'attente. A la place, le nom de l'exit de charge de travail de cluster est défini par l'attribut de gestionnaire de files d'attente `ClusterWorkloadExit`.

L'exit MQ\_CLUSTER\_WORKLOAD\_EXIT est pris en charge sur toutes les plateformes.

## Syntaxe

```
MQ_CLUSTER_WORKLOAD_EXIT (ExitParms)
```

### Référence associée

[MQXCLWLN -Navigation dans les enregistrements de charge de travail de cluster](#)

L'appel MQXCLWLN permet de naviguer dans les chaînes des enregistrements MQWDR, MQWQR et MQWCR stockés dans le cache du cluster.

[MQWXP -Structure des paramètres d'exit de charge de travail de cluster](#)

Le tableau suivant récapitule les zones de la structure de paramètres d'exit de charge de travail MQWXP -Cluster.

#### MQWDR-Structure d'enregistrement de destination de charge de travail de cluster

Le tableau suivant récapitule les zones de la structure d'enregistrement de destination de charge de travail MQWDR -Cluster.

#### MQWQR -Structure d'enregistrement de file d'attente de charge de travail de cluster

Le tableau suivant récapitule les zones de la structure d'enregistrement de file d'attente de charge de travail de cluster MQWQR .

#### MQWCR -Structure d'enregistrement de cluster de charge de travail de cluster

Le tableau suivant récapitule les zones de la structure d'enregistrement de charge de travail du cluster MQWCR .

### ***paramètres pour MQ\_CLUSTER\_WORKLOAD\_EXIT***

Description des paramètres dans l'appel MQ\_CLUSTER\_WORKLOAD\_EXIT .

#### ***ExitParms ( MQWXP ) -entrée / sortie***

Bloc de paramètres d'exit.

- L'exit définit les informations dans MQWXP pour indiquer comment gérer la charge de travail.

#### **Référence associée**

##### Remarques sur l'utilisation

La fonction exécutée par l'exit de charge de travail de cluster est définie par le fournisseur de l'exit. Toutefois, l'exit doit être conforme aux règles définies dans le bloc de contrôle associé MQWXP.

##### Appels de langue pour MQ\_CLUSTER\_WORKLOAD\_EXIT

MQ\_CLUSTER\_WORKLOAD\_EXIT prend en charge deux langages, C et High Level Assembler.

#### ***Remarques sur l'utilisation***

La fonction exécutée par l'exit de charge de travail de cluster est définie par le fournisseur de l'exit. Toutefois, l'exit doit être conforme aux règles définies dans le bloc de contrôle associé MQWXP.

Aucun point d'entrée appelé MQ\_CLUSTER\_WORKLOAD\_EXIT n'est fourni par le gestionnaire de files d'attente. Toutefois, un typedef est fourni pour le nom MQ\_CLUSTER\_WORKLOAD\_EXIT dans le langage de programmation C. Utilisez typedef pour déclarer l'exit écrit par l'utilisateur, afin de vous assurer que les paramètres sont corrects.

#### **Référence associée**

##### paramètres pour MQ\_CLUSTER\_WORKLOAD\_EXIT

Description des paramètres dans l'appel MQ\_CLUSTER\_WORKLOAD\_EXIT .

##### Appels de langue pour MQ\_CLUSTER\_WORKLOAD\_EXIT

MQ\_CLUSTER\_WORKLOAD\_EXIT prend en charge deux langages, C et High Level Assembler.

#### ***Appels de langue pour MQ\_CLUSTER\_WORKLOAD\_EXIT***

MQ\_CLUSTER\_WORKLOAD\_EXIT prend en charge deux langages, C et High Level Assembler.

### **Appel C**

```
MQ_CLUSTER_WORKLOAD_EXIT (&ExitParms);
```

Remplacez `MQ_CLUSTER_WORKLOAD_EXIT` par le nom de votre fonction d'exit de charge de travail de cluster.

Déclarez les paramètres **MQ\_CLUSTER\_WORKLOAD\_EXIT** comme suit:

```
MQWXP ExitParms; /* Exit parameter block */
```

## Appel d'assembleur de haut niveau

```
CALL EXITNAME,(EXITPARMS)
```

Déclarez les paramètres comme suit :

EXITPARMS	CMQWXP	Exit parameter block
-----------	--------	----------------------

### Référence associée

paramètres pour MQ\_CLUSTER\_WORKLOAD\_EXIT

Description des paramètres dans l'appel MQ\_CLUSTER\_WORKLOAD\_EXIT .

### Remarques sur l'utilisation

La fonction exécutée par l'exit de charge de travail de cluster est définie par le fournisseur de l'exit. Toutefois, l'exit doit être conforme aux règles définies dans le bloc de contrôle associé MQWXP.

## MQXCLWLN -Navigation dans les enregistrements de charge de travail de cluster

L'appel MQXCLWLN permet de naviguer dans les chaînes des enregistrements MQWDR, MQWQR et MQWCR stockés dans le cache du cluster.

Le cache du cluster est une zone de mémoire principale utilisée pour stocker les informations relatives au cluster.

Si le cache du cluster est statique, sa taille est fixe. Si vous lui affectez la valeur dynamique, le cache du cluster peut être étendu en fonction des besoins.

Définissez le type de cache de cluster sur STATIC ou DYNAMIC à l'aide d'un paramètre système ou d'une macro.

- **Multi** Utilisez le paramètre système ClusterCacheType sur [Multiplateformes](#).
- **z/OS** Utilisez le paramètre CLCACHE dans la macro CSQ6SYSP sous z/OS.

## Syntaxe

```
MQXCLWLN (ExitParms, CurrentRecord, NextOffset, NextRecord, Compcode, Reason)
```

### Référence associée

MQ\_CLUSTER\_WORKLOAD\_EXIT -Description de l'appel

L'exit de charge de travail de cluster est appelé par le gestionnaire de files d'attente pour acheminer un message vers un gestionnaire de files d'attente disponible.

[MQWXP -Structure des paramètres d'exit de charge de travail de cluster](#)

Le tableau suivant récapitule les zones de la structure de paramètres d'exit de charge de travail MQWXP -Cluster.

[MQWDR-Structure d'enregistrement de destination de charge de travail de cluster](#)

Le tableau suivant récapitule les zones de la structure d'enregistrement de destination de charge de travail MQWDR -Cluster.

[MQWQR -Structure d'enregistrement de file d'attente de charge de travail de cluster](#)

Le tableau suivant récapitule les zones de la structure d'enregistrement de file d'attente de charge de travail de cluster MQWQR .

[MQWCR -Structure d'enregistrement de cluster de charge de travail de cluster](#)

Le tableau suivant récapitule les zones de la structure d'enregistrement de charge de travail du cluster MQWCR .

## **Paramètres de MQXCLWLN -Navigation dans les enregistrements de charge de travail de cluster**

Description des paramètres dans l'appel MQXCLWLN .

### **ExitParms ( MQWXP ) -entrée / sortie**

Bloc de paramètres d'exit.

Cette structure contient des informations relatives à l'appel de l'exit. L'exit définit des informations dans cette structure pour indiquer comment gérer la charge de travail.

### **CurrentRecord ( MQPTR ) -entrée**

Adresse de l'enregistrement en cours.

Cette structure contient des informations relatives à l'adresse de l'enregistrement en cours d'examen par la sortie. L'enregistrement doit être de l'un des types suivants:

- Enregistrement de destination de charge de travail de cluster ( MQWDR )
- Enregistrement de file d'attente de charge de travail de cluster ( MQWQR )
- Enregistrement de cluster de charge de travail de cluster ( MQWCR )

### **NextOffset ( MQLONG ) -entrée**

Décalage de l'enregistrement suivant.

Cette structure contient des informations relatives au décalage de l'enregistrement ou de la structure suivante. *NextOffset* est la valeur de la zone de décalage appropriée dans l'enregistrement en cours et doit être l'une des zones suivantes:

- ZoneChannelDefOffset dans MQWDR
- ZoneClusterRecOffset dans MQWDR
- ZoneClusterRecOffset dans MQWQR
- ZoneClusterRecDécalage dans MQWCR

### **NextRecord ( MQPTR ) -sortie**

Adresse de l'enregistrement ou de la structure suivant (s).

Cette structure contient des informations relatives à l'adresse de l'enregistrement ou de la structure suivante. Si *CurrentRecord* est l'adresse d'un MQWDRet *NextOffset* est la valeur de la zone ChannelDefOffset , *NextRecord* est l'adresse de la structure de définition de canal ( MQCD ).

S'il n'y a pas d'enregistrement ou de structure suivant, le gestionnaire de files d'attente définit *NextRecord* sur le pointeur null et l'appel renvoie le code achèvement MQCC\_WARNING et le code anomalie MQRC\_NO\_RECORD\_AVAILABLE.

### **CompCode ( MQLONG ) -sortie**

Code achèvement.

Le code achèvement comporte l'une des valeurs suivantes:

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_WARNING**

Avertissement (achèvement partiel).

#### **MQCC\_FAILED**

Echec de l'appel.

### **Raison ( MQLONG ) -sortie**

Code anomalie qualifiant CompCode

Si CompCode est MQCC\_OK:

#### **MQRC\_NONE**

( 0, X'0000' )

Aucune raison de signaler.

Si *CompCode* est MQCC\_WARNING:

**MQRC\_NO\_RECORD\_AVAILABLE**  
( 2359, X'0937' )

Aucun enregistrement disponible. Un appel MQXCLWLN a été émis à partir d'un exit de charge de travail de cluster pour obtenir l'adresse de l'enregistrement suivant dans la chaîne. L'enregistrement en cours est le dernier enregistrement de la chaîne. Action corrective: aucune.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_CURRENT\_RECORD\_ERROR**  
( 2357, X'0935' )

Paramètre **CurrentRecord** incorrect. Un appel MQXCLWLN a été émis à partir d'un exit de charge de travail de cluster pour obtenir l'adresse de l'enregistrement suivant dans la chaîne. L'adresse spécifiée par le paramètre **CurrentRecord** n'est pas l'adresse d'un enregistrement valide.

**CurrentRecord** doit être l'adresse d'un enregistrement de destination, MQWDR, d'un enregistrement de file d'attente ( MQWQR ) ou d'un enregistrement de cluster ( MQWCR ) résidant dans le cache du cluster. Action corrective: Vérifiez que l'exit de charge de travail du cluster transmet l'adresse d'un enregistrement valide résidant dans le cache du cluster.

**MQRC\_ENVIRONMENT\_ERROR**  
( 2012, X'07DC' )

Appel non valide dans l'environnement. Un appel MQXCLWLN a été émis, mais pas à partir d'un exit de charge de travail de cluster.

**MQRC\_NEXT\_OFFSET\_ERROR**  
( 2358, X'0936' )

Paramètre **NextOffset** incorrect. Un appel MQXCLWLN a été émis à partir d'un exit de charge de travail de cluster pour obtenir l'adresse de l'enregistrement suivant dans la chaîne. Le décalage spécifié par le paramètre **NextOffset** n'est pas valide. **NextOffset** doit être la valeur de l'une des zones suivantes:

- ZoneChannelDefOffset dans MQWDR
- Zone ClusterRecOffset dans MQWDR
- Zone ClusterRecOffset dans MQWQR
- ZoneClusterRecDécalage dans MQWCR

Action corrective: Vérifiez que la valeur spécifiée pour le paramètre **NextOffset** correspond à la valeur de l'une des zones répertoriées précédemment.

**MQRC\_NEXT\_RECORD\_ERROR**  
( 2361, X'0939' )

Paramètre **NextRecord** incorrect.

**MQRC\_WXP\_ERROR**  
( 2356, X'0934' )

Structure de paramètre d'exit de charge de travail incorrecte. Un appel MQXCLWLN a été émis à partir d'un exit de charge de travail de cluster pour obtenir l'adresse de l'enregistrement suivant dans la chaîne. La structure de paramètre d'exit de charge de travail **ExitParms** n'est pas valide pour l'une des raisons suivantes:

- Le pointeur de paramètre n'est pas valide. Il n'est pas toujours possible de détecter des pointeurs de paramètre non valides ; s'ils ne sont pas détectés, des résultats imprévisibles se produisent.
- La zone StrucId n'est pas MQWXP\_STRUC\_ID.
- La zone Version n'est pas MQWXP\_VERSION\_2.
- La zone Contexte ne contient pas la valeur transmise à l'exit par le gestionnaire de files d'attente.

Action corrective: Vérifiez que le paramètre spécifié pour **ExitParms** correspond à la structure MQWXP qui a été transmise à l'exit lors de son appel.



## Référence associée

[Remarques sur l'utilisation de MQXCLWLN-Navigation dans les enregistrements de charge de travail de cluster](#)

Utilisez MQXCLWLN pour naviguer dans les enregistrements de cluster, même si le cache est statique.

Appels de langue de MQXCLWLN

MQXCLWLN prend en charge deux langages, C et High Level Assembler.

## Remarques sur l'utilisation de MQXCLWLN-Navigation dans les enregistrements de charge de travail de cluster

Utilisez MQXCLWLN pour naviguer dans les enregistrements de cluster, même si le cache est statique.

Si le cache du cluster est dynamique, l'appel MQXCLWLN doit être utilisé pour naviguer dans les enregistrements. L'exit se termine de manière anormale si une simple arithmétique de pointeur et de décalage est utilisée pour naviguer dans les enregistrements.

Si le cache de cluster est statique, il n'est pas nécessaire d'utiliser MQXCLWLN pour naviguer dans les enregistrements. Utilisez généralement MQXCLWLN même lorsque le cache est statique. Vous pouvez ensuite modifier le cache du cluster pour qu'il soit dynamique sans qu'il soit nécessaire de modifier l'exit de charge de travail.

## Référence associée

[Paramètres de MQXCLWLN -Navigation dans les enregistrements de charge de travail de cluster](#)

Description des paramètres dans l'appel MQXCLWLN .

Appels de langue de MQXCLWLN

MQXCLWLN prend en charge deux langages, C et High Level Assembler.

## Appels de langue de MQXCLWLN

MQXCLWLN prend en charge deux langages, C et High Level Assembler.

## Appel C

```
MQXCLWLN (&ExitParms, CurrentRecord, NextOffset, &NextRecord, &CompCode, &Reason) ;
```

Déclarez les paramètres comme suit :

```
typedef struct tagMQXCLWLN {
MQWXP   ExitParms;      /* Exit parameter block */
MQPTR   CurrentRecord; /* Address of current record*/
MQLONG  NextOffset;    /* Offset of next record */
MQPTR   NextRecord;    /* Address of next record or structure */
MQLONG  CompCode;     /* Completion code */
MQLONG  Reason;       /* Reason code qualifying CompCode */
}
```

## Appel d'assembleur de haut niveau

```
CALL MQXCLWLN, (CLWLEXITPARMS, CURRENTRECORD, NEXTOFFSET, NEXTRECORD, COMPCODE, REASON)
```

Déclarez les paramètres comme suit :

```
CLWLEXITPARMS CMQWXP, Cluster workload exit parameter block
CURRENTRECORD CMQWDRA, Current record
NEXTOFFSET    DS F    Next offset
NEXTRECORD    DS F    Next record
COMPCODE      DS F    Completion code
REASON        DS F    Reason code qualifying COMPCODE
```

## Référence associée

[Paramètres de MQXCLWLN -Navigation dans les enregistrements de charge de travail de cluster](#)

Description des paramètres dans l'appel MQXCLWLN .

Remarques sur l'utilisation de MQXCLWLN-Navigation dans les enregistrements de charge de travail de cluster

Utilisez MQXCLWLN pour naviguer dans les enregistrements de cluster, même si le cache est statique.

## MQWXP -Structure des paramètres d'exit de charge de travail de cluster

Le tableau suivant récapitule les zones de la structure de paramètres d'exit de charge de travail MQWXP-Cluster.

Tableau 826. Zones dans MQWXP		
Zone	Description	Page
<i>StrucId</i>	Identificateur de structure	<a href="#">StrucId</a>
<i>Version</i>	Numéro de version de structure	<a href="#">Version</a>
<i>ExitId</i>	Type de sortie	<a href="#">ExitId</a>
<i>ExitReason</i>	Raison de l'appel de l'exit	<a href="#">ExitReason</a>
<i>ExitResponse</i>	Réponse de l'exit	<a href="#">ExitResponse</a>
<i>ExitResponse2</i>	Réponse secondaire de l'exit	<a href="#">ExitResponse2</a>
<i>Feedback</i>	Code retour	<a href="#">Commentaires en retour</a>
<i>Flags</i>	Marque les valeurs. Ces indicateurs de bit sont utilisés pour indiquer des informations sur le message en cours d'insertion	<a href="#">Indicateurs</a>
<i>ExitUserArea</i>	Quitter la zone utilisateur	<a href="#">ZoneExitUser</a>
<i>ExitData</i>	Données d'exit	<a href="#">ExitData</a>
<i>MsgDescPtr</i>	Adresse du descripteur de message ( MQMD )	<a href="#">MsgDescPtr</a>
<i>MsgBufferPtr</i>	Adresse de la mémoire tampon contenant une partie ou la totalité des données de message	<a href="#">MsgBufferPtr</a>
<i>MsgBufferLength</i>	Longueur de la mémoire tampon contenant les données de message	<a href="#">MsgBufferLongueur</a>
<i>MsgLength</i>	Longueur du message complet	<a href="#">MsgLength</a>
<i>QName</i>	Nom de la file d'attente	<a href="#">QName</a>
<i>QMgrName</i>	Nom du gestionnaire de files d'attente local	<a href="#">QMgrName</a>
<i>DestinationCount</i>	Nombre de destinations possibles	<a href="#">DestinationCount</a>
<i>DestinationChosen</i>	Destination choisie	<a href="#">DestinationChosen</a>
<i>DestinationArrayPtr</i>	Adresse d'un tableau de pointeurs vers des enregistrements de destination ( MQWDR )	<a href="#">DestinationArrayPtr</a>
<i>QArrayPtr</i>	Adresse d'un tableau de pointeurs vers des enregistrements de file d'attente ( MQWQR )	<a href="#">QArrayPtr</a>
<b>Remarque :</b> Les zones restantes sont ignorées si la version est inférieure à MQWXP_VERSION_2.		
<i>CacheContext</i>	Informations contextuelles	<a href="#">CacheContext</a>
<i>CacheType</i>	Type de cache de cluster	<a href="#">CacheType</a>

Tableau 826. Zones dans MQWXP (suite)		
Zone	Description	Page
<b>Remarque :</b> Les zones restantes sont ignorées si la version est inférieure à MQWXP_VERSION_3.		
<i>CLWLMRUChannels</i>	Nombre maximal de canaux de cluster sortants actifs autorisés	<a href="#">CLWLMRUChannels</a>
<b>Remarque :</b> Les zones restantes sont ignorées si la version est inférieure à MQWXP_VERSION_4.		
<i>pEntryPoints</i>	Adresse de la structure MQIEP permettant d'effectuer des appels MQI et DCI	<a href="#">pEntryPoints</a>

La structure des paramètres d'exit de charge de travail de cluster décrit les informations transmises à l'exit de charge de travail de cluster.

La structure des paramètres d'exit de charge de travail de cluster est prise en charge sur toutes les plateformes

De plus, les structures MQWXP1, MQWXP2 et MQWXP3 sont disponibles pour la compatibilité avec les versions antérieures.

### Référence associée

[MQ\\_CLUSTER\\_WORKLOAD\\_EXIT](#) -Description de l'appel

L'exit de charge de travail de cluster est appelé par le gestionnaire de files d'attente pour acheminer un message vers un gestionnaire de files d'attente disponible.

[MQXCLWLN](#) -Navigation dans les enregistrements de charge de travail de cluster

L'appel MQXCLWLN permet de naviguer dans les chaînes des enregistrements MQWDR, MQWQRet MQWCR stockés dans le cache du cluster.

[MQWDR](#) -Structure d'enregistrement de destination de charge de travail de cluster

Le tableau suivant récapitule les zones de la structure d'enregistrement de destination de charge de travail MQWDR -Cluster.

[MQWQR](#) -Structure d'enregistrement de file d'attente de charge de travail de cluster

Le tableau suivant récapitule les zones de la structure d'enregistrement de file d'attente de charge de travail de cluster MQWQR .

[MQWCR](#) -Structure d'enregistrement de cluster de charge de travail de cluster

Le tableau suivant récapitule les zones de la structure d'enregistrement de charge de travail du cluster MQWCR .

### Zones dans MQWXP -Structure des paramètres d'exit de charge de travail de cluster

Description des zones de la structure de paramètres de l'exit de charge de travail de cluster MQWXP

#### StrucId (MQCHAR4)-entrée

Identificateur de structure de la structure du paramètre d'exit de charge de travail de cluster.

- La valeur StrucId est MQWXP\_STRUC\_ID.
- Pour le langage de programmation C, la constante MQWXP\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQWXP\_STRUC\_ID. Il s'agit d'un tableau de caractères au lieu d'une chaîne.

#### Version (MQLONG)-entrée

Indique le numéro de version de la structure. Version prend l'une des valeurs suivantes:

##### MQWXP\_VERSION\_1



Structure des paramètres d'exit de charge de travail de cluster Version-1 .

MQWXP\_VERSION\_1 est pris en charge dans tous les environnements.

##### MQWXP\_VERSION\_2

Structure des paramètres d'exit de charge de travail de cluster Version-2 .

MQWXP\_VERSION\_2 est pris en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows

### **MQWXP\_VERSION\_3**

Structure des paramètres d'exit de charge de travail de cluster Version-3 .

MQWXP\_VERSION\_3 est pris en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows

### **MQWXP\_VERSION\_4**

Structure des paramètres d'exit de charge de travail de cluster Version-4 .

MQWXP\_VERSION\_4 est pris en charge dans les environnements suivants:

-  AIX
-  IBM i
-  Linux
-  Windows

### **MQWXP\_CURRENT\_VERSION**

Version actuelle de la structure des paramètres d'exit de charge de travail de cluster.

### **ExitId (MQLONG)-entrée**

Indique le type d'exit appelé. L'exit de charge de travail de cluster est le seul exit pris en charge.

- La valeur ExitId doit être MQXT\_CLUSTER\_WORKLOAD\_EXIT

### **ExitReason (MQLONG)-entrée**

Indique la raison pour laquelle l'exit de charge de travail du cluster a été appelé. ExitReason prend l'une des valeurs suivantes:

#### **MQXR\_INIT**

Indique que l'exit est appelé pour la première fois.

Obtenez et initialisez les ressources dont l'exit peut avoir besoin, telles que la mémoire principale.

#### **MQXR\_TERM**

Indique que l'exit est sur le point d'être arrêté.

Libérez toutes les ressources que l'exit a pu acquérir depuis qu'il a été initialisé, comme la mémoire principale.

#### **MQXR\_CLWL\_OPEN**

Appelé par MQOPEN.

#### **MQXR\_CLWL\_PUT**

Appelé par MQPUT ou MQPUT1.

#### **MQXR\_CLWL\_MOVE**

Appelé par MCA lorsque l'état du canal a changé.

#### **MQXR\_CLWL\_REPOS**

Appelé par MQPUT ou MQPUT1 pour un message PCF de gestionnaire de référentiel.

### **MQXR\_CLWL\_REPOS\_MOVE**

Appelé par MCA pour un message PCF de gestionnaire de référentiel si l'état du canal a changé.

### **ExitResponse (MQLONG)-sortie**

Définissez ExitResponse pour indiquer si le traitement du message se poursuit. Il doit s'agir de l'une des valeurs suivantes:

#### **MQXCC\_OK**

Poursuivez le traitement du message normalement.

- DestinationChosen identifie la destination à laquelle le message doit être envoyé.

#### **MQXCC\_SUPPRESS\_FUNCTION**

Interrompez le traitement du message.

- Les actions effectuées par le gestionnaire de files d'attente dépendent de la raison pour laquelle l'exit a été appelé:

ExitReason	Action prise
– MQXR_CLWL_OPEN – MQXR_CLWL_REPOS – MQXR_CLWL_PUT	L'appel MQOPEN, MQPUT ou MQPUT1 échoue avec le code achèvement MQCC_FAILED et le code anomalie MQRC_STOPPED_BY_CLUSTER_EXIT.
– MQXR_CLWL_MOVE – MQXR_CLWL_REPOS_MOVE	Le message est placé dans la file d'attente des messages non livrés.

#### **MQXCC\_SUPPRESS\_EXIT**

Poursuivez le traitement du message en cours normalement. N'appellez pas à nouveau l'exit tant que le gestionnaire de files d'attente n'est pas arrêté.

Le gestionnaire de files d'attente traite les messages suivants comme si l'attribut de gestionnaire de files d'attente ClusterWorkloadExit est vide. DestinationChosen identifie la destination à laquelle le message en cours est envoyé.

#### **Toute autre valeur**

Traitez le message comme si MQXCC\_SUPPRESS\_FUNCTION est spécifié.

### **ExitResponse2 (MQLONG)-entrée / sortie**

Définissez ExitResponse2 pour fournir plus d'informations au gestionnaire de files d'attente.

- MQXR2\_STATIC\_CACHE est la valeur par défaut et est définie à l'entrée de l'exit.
- Lorsque ExitReason a la valeur MQXR\_INIT, l'exit peut définir l'une des valeurs suivantes dans ExitResponse2:

#### **MQXR2\_STATIC\_CACHE**

L'exit requiert un cache de cluster statique.

- Si le cache du cluster est statique, l'exit n'a pas besoin d'utiliser l'appel MQXCLWLN pour naviguer dans les chaînes d'enregistrements du cache du cluster.
- Si le cache du cluster est dynamique, l'exit ne peut pas naviguer correctement dans les enregistrements du cache.

**Remarque :** Le gestionnaire de files d'attente traite le retour de l'appel MQXR\_INIT comme si l'exit avait renvoyé MQXCC\_SUPPRESS\_EXIT dans la zone ExitResponse .

#### **MQXR2\_DYNAMIC\_CACHE**

L'exit peut fonctionner avec un cache statique ou dynamique.

- Si l'exit renvoie cette valeur, il doit utiliser l'appel MQXCLWLN pour naviguer dans les chaînes d'enregistrements du cache du cluster.

**Feedback (MQLONG)-entrée**

Zone réservée. La valeur est zéro.

**Indicateurs (MQLONG)-entrée**

Indique des informations sur le message en cours d'insertion.

- La valeur de `Flags` est `MQWXP_PUT_BY_CLUSTER_CHL`. Le message provient d'un canal de cluster, plutôt que d'un canal local ou d'un canal non-cluster. En d'autres termes, le message provient d'un autre gestionnaire de files d'attente de cluster.

**Réservé (MQLONG)-entrée**

Zone réservée. La valeur est zéro.

**ExitUserArea (MQBYTE16)-entrée / sortie**

Définissez `ExitUserArea` pour communiquer entre les appels à l'exit.

- La zone `ExitUser` est initialisée à zéro binaire avant le premier appel de l'exit. Toutes les modifications apportées à cette zone par l'exit sont conservées entre les appels de l'exit qui se produisent entre l'appel `MQCONN` et l'appel `MQDISC` correspondant. La zone est réinitialisée à zéro binaire lorsque l'appel `MQDISC` est effectué.
- Le premier appel de l'exit est indiqué par la zone `ExitReason` ayant la valeur `MQXR_INIT`.
- Les constantes suivantes sont définies:

**MQXUA\_NONE -chaîne****MQXUA\_NONE\_ARRAY -tableau de caractères**

Aucune information utilisateur. Les deux constantes sont égales à zéro binaire pour la longueur de la zone.

**MQ\_EXIT\_USER\_AREA\_LENGTH**

Longueur de la zone `ExitUser`.

**ExitData (MQCHAR32)-entrée**

Valeur de l'attribut de gestionnaire de files d'attente `ClusterWorkloadDonnées`. Si aucune valeur n'a été définie pour cet attribut, la zone ne comporte que des blancs.

- La longueur de `ExitData` est donnée par `MQ_EXIT_DATA_LENGTH`.

**MsgDescPtr (PMQMD)-entrée**

Adresse d'une copie du descripteur de message (MQMD) pour le message en cours de traitement.

- Toutes les modifications apportées au descripteur de message par l'exit sont ignorées par le gestionnaire de files d'attente.
- Si `ExitReason` a l'une des valeurs suivantes `MsgDescPtr` est défini sur le pointeur null et aucun descripteur de message n'est transmis à l'exit:
  - `MQXR_INIT`
  - `MQXR_TERM`
  - `MQXR_CLWL_OPEN`

**MsgBufferPtr (PMQVOID)-Entrée**

Adresse d'une mémoire tampon contenant une copie du premier `MsgBufferLongueur` octets des données de message.

- Toutes les modifications apportées aux données de message par l'exit sont ignorées par le gestionnaire de files d'attente.
- Aucune donnée de message n'est transmise à l'exit lorsque:
  - `MsgDescPtr` est le pointeur null.
  - Le message ne contient pas de données.
  - L'attribut de gestionnaire de files d'attente `ClusterWorkloadLongueur` est égal à zéro.Dans ces cas, `MsgBufferPtr` est le pointeur null.

**MsgBufferLength (MQLONG)-entrée**

Longueur de la mémoire tampon contenant les données de message transmises à l'exit.

- La longueur est contrôlée par l'attribut de gestionnaire de files d'attente `ClusterWorkloadLongueur`.
- La longueur peut être inférieure à la longueur du message complet. Voir `MsgLength`.

**MsgLength (MQLONG)-entrée**

Longueur du message complet transmis à l'exit.

- `MsgBufferLa` longueur peut être inférieure à la longueur du message complet.
- `MsgLength` est égal à zéro si `ExitReason` est défini sur `MQXR_INIT`, `MQXR_TERMou` `MQXR_CLWL_OPEN`.

**QName (MQCHAR48)-entrée**

Nom de la file d'attente de destination. La file d'attente est une file d'attente de cluster.

- La longueur de `QName` est `MQ_Q_NAME_LENGTH`.

**QMgrName (MQCHAR48)-entrée**

Nom du gestionnaire de files d'attente local qui a appelé l'exit de charge de travail de cluster.

- La longueur de `QMgrName` est `MQ_Q_MGR_NAME_LENGTH`.

**DestinationCount (MQLONG)-entrée**

Nombre de destinations possibles. Les destinations sont des instances de la file d'attente de destination et sont décrites par les enregistrements de destination.

- Un enregistrement de destination est une structure `MQWDR`. Il existe une structure pour chaque route possible vers chaque instance de la file d'attente.
- Les structures `MQWDR` sont traitées par un tableau de pointeurs. Voir `DestinationArrayPtr`.

**DestinationChosen (MQLONG)-entrée/sortie**

Destination choisie.

- Numéro de la structure `MQWDR` qui identifie la route et l'instance de file d'attente où le message doit être envoyé.
- La valeur est comprise entre 1 et `DestinationCount`.
- En entrée de l'exit, `DestinationChosen` indique la route et l'instance de file d'attente sélectionnées par le gestionnaire de files d'attente. L'exit peut accepter ce choix ou choisir une autre route et une autre instance de file d'attente.
- La valeur définie par l'exit doit être comprise entre 1 et `DestinationCount`. Si une autre valeur est renvoyée, le gestionnaire de files d'attente utilise la valeur de `DestinationChosen` en entrée de l'exit.

**DestinationArrayPtr (PPMQWDR)-entrée**

Adresse d'un tableau de pointeurs vers des enregistrements de destination (`MQWDR`).

- Il existe des enregistrements de destination `DestinationCount`.

**QArrayPtr (PPMQQR)-entrée**

Adresse d'un tableau de pointeurs vers des enregistrements de file d'attente (`MQWQR`).

- Si des enregistrements de file d'attente sont disponibles, ils sont `DestinationCount`.
- Si aucun enregistrement de file d'attente n'est disponible, `QArrayPtr` est le pointeur null.

**Remarque :** `QArrayPtr` peut être le pointeur null même si `DestinationCount` est supérieur à zéro.

**CacheContext (MQPTR): Version 2-input**

La zone `CacheContext` est réservée au gestionnaire de files d'attente. L'exit ne doit pas modifier la valeur de cette zone.

### CacheType (MQLONG): Version 2-input

Le cache de cluster possède l'un des types suivants:

#### MQCLCT\_STATIC

Le cache est statique.

- La taille du cache est fixe et ne peut pas augmenter lorsque le gestionnaire de files d'attente fonctionne.
- Vous n'avez pas besoin d'utiliser l'appel MQXCLWLN pour naviguer dans les enregistrements de ce type de cache.

#### MQCLCT\_DYNAMIC

Le cache est dynamique.

- La taille de la mémoire cache peut augmenter afin de prendre en charge les informations de cluster variables.
- Vous devez utiliser l'appel MQXCLWLN pour naviguer dans les enregistrements de ce type de cache.

### CLWLMRUChannels (MQLONG): Version 3-entrée

Indique le nombre maximal de canaux de cluster sortants actifs à utiliser par l'algorithme de choix de la charge de travail du cluster.

- CLWLMRUChannels est une valeur comprise entre 1 et 999 999 999.

### pEntryPoints (PMQIEP): Version 4

Adresse d'une structure MQIEP via laquelle les appels MQI et DCI peuvent être effectués.

### Référence associée

Valeurs initiales et déclarations de langue pour MQWXP

Valeurs initiales et déclarations de langage C et High Level Assembler pour MQWXP -Structure des paramètres d'exit de charge de travail de cluster.

### Valeurs initiales et déclarations de langue pour MQWXP

Valeurs initiales et déclarations de langage C et High Level Assembler pour MQWXP -Structure des paramètres d'exit de charge de travail de cluster.

Nom de zone	Nom de la constante	Valeur de la constante
<i>StrucId</i>	MQWXP_STRUC_ID	'WXP~'
<i>Version</i>	MQWXP_VERSION_2	2
<i>ExitId</i>	Aucun	0
<i>ExitReason</i>	MQXCC_OK	0
<i>ExitResponse</i>	Aucun	0
<i>ExitResponse2</i>	Aucun	0
<i>Flags</i>	Aucun	0
<i>ExitUserArea</i>	{MQXUA_NONE_ARRAY}	0
<i>ExitData</i>	Aucun	''
<i>MsgDescPtr</i>	Aucun	NULL
<i>MsgBufferPtr</i>	Aucun	NULL
<i>MsgBufferLength</i>	Aucun	0
<i>MsgBufferPtr</i>	Aucun	0



Tableau 828. Zones dans MQWXP (suite)

Nom de zone	Nom de la constante	Valeur de la constante
<i>QName</i>	Aucun	" "
<i>QMgrName</i>	Aucun	" "
<i>DestinationCount</i>	Aucun	0
<i>DestinationChosen</i>	Aucun	0
<i>DestinationArrayPtr</i>	Aucun	NULL
<i>QArrayPtr</i>	Aucun	NULL
<i>CacheContext</i>	Aucun	NULL
<i>CacheType</i>	MQCLCT_DYNAMIC	1
<i>CLWLMRUChannels</i>	Aucun	0
<i>pEntryPoints</i>	Aucun	NULL

**Remarques :**

1. Le symbole ~ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable de macro MQWXP\_DEFAULT contient les valeurs par défaut. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQWDR MyWXP = {MQWXP_DEFAULT};
```

**Déclaration C**

```
typedef struct tagMQWXP {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    ExitId;           /* Type of exit */
    MQLONG    ExitReason;       /* Reason for invoking exit */
    MQLONG    ExitResponse;     /* Response from exit */
    MQLONG    ExitResponse2;    /* Reserved */
    MQLONG    Feedback;        /* Reserved */
    MQLONG    Flags;            /* Flags */
    MQBYTE16  ExitUserArea;     /* Exit user area */
    MQCHAR32  ExitData;         /* Exit data */
    PMQMD     MsgDescPtr;       /* Address of message descriptor */
    PMQVOID   MsgBufferPtr;     /* Address of buffer containing some
                                or all of the message data */
    MQLONG    MsgBufferLength;  /* Length of buffer containing message
                                data */
    MQLONG    MsgLength;        /* Length of complete message */
    MQCHAR48  QName;            /* Queue name */
    MQCHAR48  QMgrName;         /* Name of local queue manager */
    MQLONG    DestinationCount; /* Number of possible destinations */
    MQLONG    DestinationChosen; /* Destination chosen */
    PPMQWDR   DestinationArrayPtr; /* Address of an array of pointers to
                                destination records */
    PPMQWQR   QArrayPtr;       /* Address of an array of pointers to
                                queue records */
    /* version 1 */
    MQPTR     CacheContext;     /* Context information */
    MQLONG    CacheType;        /* Type of cluster cache */
    /* version 2 */
    MQLONG    CLWLMRUChannels;  /* Maximum number of most recently
                                used cluster channels */
    /* version 3 */
    PMQIEP    pEntryPoints;     /* Address of the MQIEP structure */
};
```

```

/* version 4 */
};

```

## High Level Assembler

```

MQWXP                                DSECT
MQWXP_STRUCID                        DS    CL4      Structure identifier
MQWXP_VERSION                        DS    F       Structure version number
MQWXP_EXITID                         DS    F       Type of exit
MQWXP_EXITREASON                     DS    F       Reason for invoking exit
MQWXP_EXITRESPONSE                   DS    F       Response from exit
MQWXP_EXITRESPONSE2                  DS    F       Reserved
MQWXP_FEEDBACK                       DS    F       Reserved
MQWXP_RESERVED                       DS    F       Reserved
MQWXP_EXITUSERAREA                   DS    XL16    Exit user area
MQWXP_EXITDATA                       DS    CL32    Exit data
MQWXP_MSGDESCPTR                     DS    F       Address of message
*                                     descriptor
MQWXP_MSGBUFFERPTR                   DS    F       Address of buffer containing
*                                     some or all of the message
*                                     data
MQWXP_MSGBUFFERLENGTH                 DS    F       Length of buffer containing
*                                     message data
MQWXP_MSGLength                       DS    F       Length of complete message
MQWXP_QNAME                           DS    CL48    Queue name
MQWXP_QMGRNAME                       DS    CL48    Name of local queue manager
MQWXP_DESTINATIONCOUNT               DS    F       Number of possible
*                                     destinations
MQWXP_DESTINATIONCHOSEN               DS    F       Destination chosen
MQWXP_DESTINATIONARRAYPTR            DS    F       Address of an array of
*                                     pointers to destination
*                                     records
MQWXP_QARRAYPTR                       DS    F       Address of an array of
*                                     pointers to queue records
MQWXP_CACHECONTEXT                   DS    F       Context information
MQWXP_CACHETYPE                       DS    F       Type of cluster cache
MQWXP_CLWLMLRUCHANNELS                DS    F       Number of most recently used
*                                     channels for workload balancing

MQWXP_LENGTH                          EQU    *-MQWXP Length of structure
MQWXP_AREA                            ORG    MQWXP
DS    CL(MQWXP_LENGTH)

```

### Référence associée

Zones dans MQWXP -Structure des paramètres d'exit de charge de travail de cluster

Description des zones de la structure de paramètres de l'exit de charge de travail de cluster MQWXP

## MQWDR-Structure d'enregistrement de destination de charge de travail de cluster

Le tableau suivant récapitule les zones de la structure d'enregistrement de destination de charge de travail MQWDR -Cluster.

Tableau 829. Zones dans MQWDR		
Zone	Description	Page
<i>StrucId</i>	Identificateur de structure	<a href="#">StrucId</a>
<i>Version</i>	Numéro de version de structure	<a href="#">Version</a>
<i>StrucLength</i>	Longueur de la structure MQWDR	<a href="#">StrucLength</a>
<i>QMgrFlags</i>	Indicateurs du gestionnaire de files d'attente	<a href="#">QMgrFlags</a>
<i>QMgrIdentif</i>	Identificateur du gestionnaire de files d'attente	<a href="#">QMgrIdentif</a>
<i>QMgrName</i>	Nom gest. de files	<a href="#">QMgrName</a>
<i>ClusterRecOffset</i>	Décalage logique du premier enregistrement de cluster ( MQWCR )	<a href="#">ClusterRec</a>

Tableau 829. Zones dans MQWDR (suite)		
Zone	Description	Page
<i>ChannelState</i>	Etat du canal	<a href="#">ChannelState</a>
<i>ChannelDefOffset</i>	Décalage logique de la structure de définition de canal ( MQCD )	<a href="#">DécalageChannelDef</a>
<b>Remarque :</b> Les zones restantes sont ignorées si la version est inférieure à MQWDR_VERSION_2.		
<i>DestSeqNumber</i>	Numéro de séquence de la destination du canal	<a href="#">DestSeqNuméro</a>
<i>DestSeqFactor</i>	Facteur de séquence de destination de canal pour la pondération	<a href="#">FacteurDestSeq</a>

La structure d'enregistrement de destination de charge de travail de cluster contient des informations relatives à l'une des destinations possibles pour le message. Il existe une structure d'enregistrement de destination de charge de travail de cluster pour chaque instance de la file d'attente de destination.

La structure d'enregistrement de destination de charge de travail de cluster est prise en charge dans tous les environnements.

En outre, les structures MQWDR1 et MQWDR2 sont disponibles pour la compatibilité avec les versions antérieures.

#### Référence associée

[MQ\\_CLUSTER\\_WORKLOAD\\_EXIT](#) -Description de l'appel

L'exit de charge de travail de cluster est appelé par le gestionnaire de files d'attente pour acheminer un message vers un gestionnaire de files d'attente disponible.

[MQXCLWLN](#) -Navigation dans les enregistrements de charge de travail de cluster

L'appel MQXCLWLN permet de naviguer dans les chaînes des enregistrements MQWDR, MQWQRet MQWCR stockés dans le cache du cluster.

[MQWXP](#) -Structure des paramètres d'exit de charge de travail de cluster

Le tableau suivant récapitule les zones de la structure de paramètres d'exit de charge de travail MQWXP -Cluster.

[MQWQR](#) -Structure d'enregistrement de file d'attente de charge de travail de cluster

Le tableau suivant récapitule les zones de la structure d'enregistrement de file d'attente de charge de travail de cluster MQWQR .

[MQWCR](#) -Structure d'enregistrement de cluster de charge de travail de cluster

Le tableau suivant récapitule les zones de la structure d'enregistrement de charge de travail du cluster MQWCR .

### **Zones dans MQWDR-Structure d'enregistrement de destination de charge de travail de cluster**

Description des paramètres dans MQWDR -Structure d'enregistrement de destination de charge de travail de cluster.

#### **StrucId ( MQCHAR4 ) -entrée**

Identificateur de structure de la structure d'enregistrement de destination de charge de travail du cluster.

- La valeur StrucId est MQWDR\_STRUC\_ID.
- Pour le langage de programmation C, la constante MQWDR\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQWDR\_STRUC\_ID. Il s'agit d'un tableau de caractères au lieu d'une chaîne.

#### **Version ( MQLONG ) -entrée**

Numéro de version de la structure. Version prend l'une des valeurs suivantes:

**MQWDR\_VERSION\_1**

Enregistrement de destination de charge de travail de cluster Version-1 .

**MQWDR\_VERSION\_2**

Enregistrement de destination de charge de travail de cluster Version-2 .

**MQWDR\_CURRENT\_VERSION**

Version actuelle de l'enregistrement de destination de charge de travail du cluster.

**StrucLength ( MQLONG ) -entrée**

Longueur de la structure MQWDR . StrucLength prend l'une des valeurs suivantes:

**MQWDR\_LENGTH\_1**

Longueur de l'enregistrement de destination de charge de travail de cluster version-1 .

**MQWDR\_LENGTH\_2**

Longueur de l'enregistrement de destination de charge de travail de cluster version-2 .

**MQWDR\_CURRENT\_LENGTH**

Longueur de la version actuelle de l'enregistrement de destination de charge de travail de cluster.

**QMgrFlags ( MQLONG ) -entrée**

Indicateurs de gestionnaire de files d'attente indiquant les propriétés du gestionnaire de files d'attente qui héberge l'instance de la file d'attente de destination décrite par la structure MQWDR .

Les indicateurs suivants sont définis:

**MQQMF\_REPOSITORY\_Q\_MGR**

La destination est un gestionnaire de files d'attente de référentiel complet.

**MQQMF\_CLUSSDR\_USER\_DEFINED**

Le canal émetteur de cluster a été défini manuellement.

**MQQMF\_CLUSSDR\_AUTO\_DEFINED**

Le canal émetteur de cluster a été défini automatiquement.

**MQQMF\_AVAILABLE**

Le gestionnaire de files d'attente de destination est disponible pour recevoir des messages.

**Autres valeurs**

D'autres indicateurs de la zone peuvent être définis par le gestionnaire de files d'attente à des fins internes.

**QMgrIdentifiant ( MQCHAR48 ) -entrée**

L'identificateur de gestionnaire de files d'attente est un identificateur unique pour le gestionnaire de files d'attente qui héberge l'instance de la file d'attente de destination décrite par la structure MQWDR .

- L'identificateur est généré par le gestionnaire de files d'attente.
- La longueur de QMgrIdentifiant est MQ\_Q\_MGR\_IDENTIFIANT\_LENGTH.

**QMgrName ( MQCHAR48 ) -entrée**

Nom du gestionnaire de files d'attente qui héberge l'instance de la file d'attente de destination décrite par la structure MQWDR .

- QMgrName peut être le nom du gestionnaire de files d'attente local, ainsi qu'un autre gestionnaire de files d'attente dans le cluster.
- La longueur de QMgrName est MQ\_Q\_MGR\_NAME\_LENGTH.

**ClusterRecClusterRec ( MQLONG ) -entrée**

Décalage logique de la première structure MQWCR qui appartient à la structure MQWDR .

- Pour les caches statiques, ClusterRecOffset est le décalage de la première structure MQWCR qui appartient à la structure MQWDR .
- Le décalage est mesuré en octets à partir du début de la structure MQWDR .
- N'utilisez pas le décalage logique pour l'arithmétique de pointeur avec des caches dynamiques. Pour obtenir l'adresse de l'enregistrement suivant, l'appel MQXCLWLN doit être utilisé.

### **ChannelState ( MQLONG ) -entrée**

Etat du canal qui lie le gestionnaire de files d'attente local au gestionnaire de files d'attente identifié par la structure MQWDR . Les valeurs suivantes sont possibles :

#### **MQCHS\_BINDING**

Le canal négocie avec le partenaire.

#### **MQCHS\_INACTIVE**

Le canal n'est pas actif.

#### **MQCHS\_INITIALIZING**

Le canal est en cours d'initialisation.

#### **MQCHS\_PAUSED**

Le canal a été mis en pause.

#### **MQCHS\_REQUESTING**

Le canal demandeur demande une connexion.

#### **MQCHS\_RETRYING**

Le canal tente à nouveau d'établir une connexion.

#### **MQCHS\_RUNNING**

Le canal est en cours de transfert ou en attente de messages.

#### **MQCHS\_STARTING**

Le canal est en attente de devenir actif.

#### **MQCHS\_STOPPING**

Le canal est en cours d'arrêt.

#### **MQCHS\_STOPPED**

Le canal s'est arrêté.

### **ChannelDefChannelDef ( MQLONG ) -entrée**

Décalage logique de la définition de canal ( MQCD ) pour le canal qui lie le gestionnaire de files d'attente local au gestionnaire de files d'attente identifié par la structure MQWDR .

- ChannelDefOffset est similaire à ClusterRecOffset
- Le décalage logique ne peut pas être utilisé dans l'arithmétique de pointeur. Pour obtenir l'adresse de l'enregistrement suivant, l'appel MQXCLWLN doit être utilisé.

### **DestSeqDestSeq ( MQLONG ) -entrée**

Facteur de séquence de destination qui permet de choisir le canal en fonction de la pondération.

- Le facteurDestSeq est utilisé avant que le gestionnaire de files d'attente ne le modifie.
- Le gestionnaire de charge de travail augmente le facteurDestSeq de manière à garantir que les messages sont distribués sur les canaux en fonction de leur poids.

### **DestSeqNombre ( MQLONG ) -entrée**

Valeur de destination du canal de cluster avant que le gestionnaire de files d'attente ne la modifie.

- Le gestionnaire de charge de travail augmente DestSeqNumber chaque fois qu'un message est inséré dans ce canal.
- Les exits de charge de travail peuvent utiliser DestSeqNumber pour déterminer le canal sur lequel un message doit être inséré.

### **Référence associée**

Valeurs initiales et déclarations de langue pour MQWDR

Valeurs initiales et déclarations C et High Level Assembler pour MQWDR -Enregistrement de destination de charge de travail de cluster.

### ***Valeurs initiales et déclarations de langue pour MQWDR***

Valeurs initiales et déclarations C et High Level Assembler pour MQWDR -Enregistrement de destination de charge de travail de cluster.

Tableau 830. Zones dans MQWDR

Nom de zone	Nom de la constante	Valeur de la constante
<i>StrucId</i>	MQWDR_STRUC_ID	'WDR~'
<i>Version</i>	MQWDR_VERSION_1	1
<i>StrucLength</i>	MQWDR_CURRENT_LENGTH <sup>3</sup>	136
<i>QMgrFlags</i>	MQWDR_NONE	0
<i>QMgrIdentif</i>	Aucun	" "
<i>QMgrName</i>	Aucun	" "
<i>ClusterRecOffset</i>	Aucun	0
<i>ChannelState</i>	Aucun	0
<i>ChannelDefOffset</i>	Aucun	0
<i>DestSeqNumber</i>	Aucun	0
<i>DestSeqFactor</i>	Aucun	0

**Remarques :**

1. Le symbole ~ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable de macro MQWDR\_DEFAULT contient les valeurs par défaut. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQWDR MyWDR = {MQWDR_DEFAULT};
```

3. Les valeurs initiales définissent intentionnellement la longueur de la structure sur la longueur de la version en cours, et non sur la version 1 de la structure.

## High Level Assembler

```
MQWDR          DSECT
MQWDR_STRUCID  DS   CL4      Structure identifier
MQWDR_VERSION  DS   F        Structure version number
MQWDR_STRUCLNTH DS   F        Length of MQWDR structure
MQWDR_QMGRFLGS DS   F        Queue manager flags
MQWDR_QMGRIDENTIF DS CL48    Queue manager identifier
MQWDR_QMGRNAME DS   CL48    Queue manager name
MQWDR_CLUSTERRECOFFSET DS   F  Offset of first cluster
*              record
MQWDR_CHANNELSTATE DS   F    Channel state
MQWDR_CHANNELDEFOFFSET DS   F  Offset of channel definition
*              structure
MQWDR_LENGTH     EQU  *-MQWDR Length of structure
MQWDR_AREA       ORG  MQWDR
                 DS   CL(MQWDR_LENGTH)
```

## Déclaration C

```
typedef struct tagMQWDR {
    MQCHAR4   StrucId;          /* Structure identifier */
    MQLONG    Version;         /* Structure version number */
    MQLONG    StrucLength;     /* Length of MQWDR structure */
    MQLONG    QMgrFlags;       /* Queue manager flags */
    MQCHAR48  QMgrIdentif;     /* Queue manager identifier */
    MQCHAR48  QMgrName;        /* Queue manager name */
    MQLONG    ClusterRecOffset; /* Offset of first cluster record */
};
```

```

MQLONG    ChannelState;      /* Channel state */
MQLONG    ChannelDefOffset; /* Offset of channel definition structure */
/* Ver:1 */
MQLONG    DestSeqNumber;    /* Cluster channel destination sequence number */
MQINT64   DestSeqFactor;    /* Cluster channel factor sequence number */
/* Ver:2 */
};

```

### Référence associée

Zones dans MQWDR-Structure d'enregistrement de destination de charge de travail de cluster  
Description des paramètres dans MQWDR -Structure d'enregistrement de destination de charge de travail de cluster.

## MQWQR -Structure d'enregistrement de file d'attente de charge de travail de cluster

Le tableau suivant récapitule les zones de la structure d'enregistrement de file d'attente de charge de travail de cluster MQWQR .

Tableau 831. Zones dans MQWQR		
Zone	Description	Page
<i>StrucId</i>	Identificateur de structure	<a href="#">StrucId</a>
<i>Version</i>	Numéro de version de structure	<a href="#">Version</a>
<i>StrucLength</i>	Longueur de la structure MQWQR	<a href="#">StrucLength</a>
<i>QFlags</i>	Indicateurs de file d'attente	<a href="#">Indicateurs de qualité (QFlags)</a>
<i>QName</i>	Nom de la file d'attente	<a href="#">QName</a>
<i>QMgrIdentifier</i>	Identificateur du gestionnaire de files d'attente	<a href="#">QMgrIdentifier</a>
<i>ClusterRecOffset</i>	Décalage du premier enregistrement de cluster (MQWCR)	<a href="#">ClusterRec</a>
<i>QType</i>	Type de file d'attente	<a href="#">QType</a>
<i>QDesc</i>	Description de file d'attente	<a href="#">QDesc</a>
<i>DefBind</i>	Liaison par défaut	<a href="#">DefBind</a>
<i>DefPersistence</i>	Persistance par défaut	<a href="#">DefPersistence</a>
<i>DefPriority</i>	Priorité par défaut	<a href="#">DefPriority</a>
<i>InhibitPut</i>	Indique si les opérations d'insertion dans la file d'attente sont autorisées	<a href="#">InhibitPut</a>
<b>Remarque :</b> Les zones restantes sont ignorées si la version est inférieure à MQWQR_VERSION_2.		
<i>CWLQueuePriority</i>	Une valeur comprise entre 0 et 9 représentant la priorité de la file d'attente	<a href="#">CLWLQueuePriority</a>
<i>CLWLQueueRank</i>	Une valeur comprise entre 0 et 9 représentant le rang de la file d'attente	<a href="#">CLWLQueueRank</a>
<b>Remarque :</b> Les zones restantes sont ignorées si la version est inférieure à MQWQR_VERSION_3.		
<i>DefPutResponse</i>	Réponse d'insertion par défaut	<a href="#">RéponseDefPut</a>

La structure d'enregistrement de la file d'attente de charge de travail du cluster contient des informations relatives à l'une des destinations possibles pour le message. Il existe une structure d'enregistrement de file d'attente de charge de travail de cluster pour chaque instance de la file d'attente de destination.

La structure d'enregistrement de file d'attente de charge de travail de cluster est prise en charge dans tous les environnements.

De plus, les structures MQWQR1 et MQWQR2 sont disponibles pour la compatibilité avec les versions antérieures.

### **Référence associée**

MQ\_CLUSTER\_WORKLOAD\_EXIT -Description de l'appel

L'exit de charge de travail de cluster est appelé par le gestionnaire de files d'attente pour acheminer un message vers un gestionnaire de files d'attente disponible.

MQXCLWLN -Navigation dans les enregistrements de charge de travail de cluster

L'appel MQXCLWLN permet de naviguer dans les chaînes des enregistrements MQWDR, MQWQR et MQWCR stockés dans le cache du cluster.

MQWXP -Structure des paramètres d'exit de charge de travail de cluster

Le tableau suivant récapitule les zones de la structure de paramètres d'exit de charge de travail MQWXP -Cluster.

MQWDR -Structure d'enregistrement de destination de charge de travail de cluster

Le tableau suivant récapitule les zones de la structure d'enregistrement de destination de charge de travail MQWDR -Cluster.

MQWCR -Structure d'enregistrement de cluster de charge de travail de cluster

Le tableau suivant récapitule les zones de la structure d'enregistrement de charge de travail du cluster MQWCR .

### ***Zones dans MQWQR -Structure d'enregistrement de file d'attente de charge de travail de cluster***

Description des zones de la structure d'enregistrement de file d'attente de charge de travail de cluster MQWQR .

#### **StrucId ( MQCHAR4 ) -entrée**

Identificateur de structure de la structure d'enregistrement de la file d'attente de charge de travail du cluster.

- La valeur StrucId est MQWQR\_STRUC\_ID.
- Pour le langage de programmation C, la constante MQWQR\_STRUC\_ID\_ARRAY est également définie. Il a la même valeur que MQWQR\_STRUC\_ID. Il s'agit d'un tableau de caractères au lieu d'une chaîne.

#### **Version ( MQLONG ) -entrée**

Numéro de version de la structure. Version prend l'une des valeurs suivantes:

##### **MQWQR\_VERSION\_1**

Enregistrement de file d'attente de charge de travail de cluster Version-1 .

##### **MQWQR\_VERSION\_2**

Enregistrement de file d'attente de charge de travail de cluster Version-2 .

##### **MQWQR\_VERSION\_3**

Enregistrement de file d'attente de charge de travail de cluster Version-3 .

##### **MQWQR\_CURRENT\_VERSION**

Version actuelle de l'enregistrement de file d'attente de charge de travail du cluster.

#### **StrucLength ( MQLONG ) -entrée**

Longueur de la structure MQWQR . StrucLength prend l'une des valeurs suivantes:

##### **MQWQR\_LENGTH\_1**

Longueur de l'enregistrement de file d'attente de charge de travail de cluster version-1 .

##### **MQWQR\_LENGTH\_2**

Longueur de l'enregistrement de file d'attente de charge de travail de cluster version-2 .



**MQWQR\_LENGTH\_3**

Longueur de l'enregistrement de file d'attente de charge de travail de cluster version-3 .

**MQWQR\_CURRENT\_LENGTH**

Longueur de la version en cours de l'enregistrement de file d'attente de charge de travail de cluster.

**QFlags ( MQLONG ) -entrée**

Les indicateurs de file d'attente indiquent les propriétés de la file d'attente. Les indicateurs suivants sont définis:

**MQQF\_LOCAL\_Q**

La destination est une file d'attente locale.

**MQQF\_CLWL\_USEQ\_ANY**

Autoriser l'utilisation des files d'attente locales et éloignées dans les insertions.

**MQQF\_CLWL\_USEQ\_LOCAL**

Autoriser uniquement les insertions de file d'attente locale.

**Autres valeurs**

D'autres indicateurs de la zone peuvent être définis par le gestionnaire de files d'attente à des fins internes.

**QName ( MQCHAR48 ) -entrée**

Nom de la file d'attente correspondant à l'une des destinations possibles du message.

- La longueur de QName est MQ\_Q\_NAME\_LENGTH.

**QMgrIdentifieur ( MQCHAR48 ) -entrée**

L'identificateur de gestionnaire de files d'attente est un identificateur unique pour le gestionnaire de files d'attente qui héberge l'instance de la file d'attente décrite par la structure MQWQR .

- L'identificateur est généré par le gestionnaire de files d'attente.
- La longueur de QMgrIdentifieur est MQ\_Q\_MGR\_IDENTIFIEUR\_LENGTH.

**ClusterRecClusterRec ( MQLONG ) -entrée**

Décalage logique de la première structure MQWCR qui appartient à la structure MQWQR .

- Pour les caches statiques, ClusterRecOffset est le décalage de la première structure MQWCR qui appartient à la structure MQWQR .
- Le décalage est mesuré en octets à partir du début de la structure MQWQR .
- N'utilisez pas le décalage logique pour l'arithmétique de pointeur avec les caches dynamiques. Pour obtenir l'adresse de l'enregistrement suivant, l'appel MQXCLWLN doit être utilisé.

**QType ( MQLONG ) -entrée**

Type de la file d'attente de destination. Les valeurs suivantes sont possibles :

**MQCQT\_LOCAL\_Q**

File d'attente locale.

**MQCQT\_ALIAS\_Q**

File d'attente alias.

**MQCQT\_REMOTE\_Q**

File d'attente éloignée.

**MQCQT\_Q\_MGR\_ALIAS**

Alias du gestionnaire de files d'attente.

**QDesc ( MQCHAR64 ) -entrée**

Attribut de file d'attente de description de file d'attente défini sur le gestionnaire de files d'attente qui héberge l'instance de la file d'attente de destination décrite par la structure MQWQR .

- La longueur de QDesc est MQ\_Q\_DESC\_LENGTH.

**DefBind ( MQLONG ) -entrée**

Attribut de file d'attente de liaison par défaut défini sur le gestionnaire de files d'attente qui héberge l'instance de la file d'attente de destination décrite par la structure MQWQR . MQBND\_BIND\_ON\_OPEN ou MQBND\_BIND\_ON\_GROUP doit être spécifié lors de l'utilisation de groupes avec des clusters. Les valeurs suivantes sont possibles:

**MQBND\_BIND\_ON\_OPEN**

Liaison corrigée par l'appel MQOPEN .

**MQBND\_BIND\_NOT\_FIXED**

Liaison non fixe.

**MQBND\_BIND\_ON\_GROUP**

Permet à une application de demander qu'un groupe de messages soit alloué à la même instance de destination.

**DefPersistence ( MQLONG ) -entrée**

Attribut de file d'attente de persistance des messages par défaut défini sur le gestionnaire de files d'attente qui héberge l'instance de la file d'attente de destination décrite par la structure MQWQR . Les valeurs suivantes sont possibles :

**MQPER\_PERSISTENT**

Le message est persistant.

**MQPER\_NOT\_PERSISTENT**

Le message n'est pas persistant.

**DefPriority ( MQLONG ) -entrée**

Attribut de file d'attente de priorité de message par défaut défini sur le gestionnaire de files d'attente qui héberge l'instance de la file d'attente de destination décrite par la structure MQWQR . La plage de priorité est comprise entre 0 et MaxPriority.

- 0 est la priorité la plus basse.
- MaxPriority est l'attribut de gestionnaire de files d'attente du gestionnaire de files d'attente qui héberge cette instance de la file d'attente de destination.

**InhibitPut ( MQLONG ) -entrée**

Attribut de file d'attente avec insertion interdite défini sur le gestionnaire de files d'attente qui héberge l'instance de la file d'attente de destination décrite par la structure MQWQR . Les valeurs suivantes sont possibles :

**MQQA\_PUT\_INHIBITED**

Les opérations d'insertion sont interdites.

**MQQA\_PUT\_ALLOWED**

Les opérations d'insertion sont autorisées.

**CLWLQueuePriority ( MQLONG ) -entrée**

Attribut de priorité de file d'attente de charge de travail de cluster défini sur le gestionnaire de files d'attente qui héberge l'instance de la file d'attente de destination décrite par la structure MQWQR .

**CLWLQueueRank ( MQLONG ) -entrée**

Rang de la file d'attente de charge de travail du cluster défini sur le gestionnaire de files d'attente qui héberge l'instance de la file d'attente de destination décrite par la structure MQWQR .

**RéponseDefPut ( MQLONG ) -entrée**

Attribut de file d'attente de réponses d'insertion par défaut défini sur le gestionnaire de files d'attente qui héberge l'instance de la file d'attente de destination décrite par la structure MQWQR . Les valeurs suivantes sont possibles :

**MQPRT\_SYNC\_RESPONSE**

Réponse synchrone aux appels MQPUT ou MQPUT1 .

**MQPRT\_ASYNC\_RESPONSE**

Réponse asynchrone aux appels MQPUT ou MQPUT1 .

## Référence associée

Valeurs initiales et déclarations de langage pour MQWQR

Valeurs initiales et déclarations C et High Level Assembler pour MQWQR -Enregistrement de file d'attente de charge de travail de cluster.

## Valeurs initiales et déclarations de langage pour MQWQR

Valeurs initiales et déclarations C et High Level Assembler pour MQWQR -Enregistrement de file d'attente de charge de travail de cluster.

Tableau 832. Zones dans MQWQR		
Nom de zone	Nom de la constante	Valeur de la constante
<i>StrucId</i>	MQWQR_STRUC_ID_ARRAY	'WQR↵'
<i>Version</i>	MQWQR_VERSION_1	1
<i>StrucLength</i>	MQWQR_CURRENT_LENGTH <sup>3</sup>	212
<i>QFlags</i>	Aucun	0
<i>QName</i>	Aucun	" "
<i>QMgrIdentifieur</i>	Aucun	" "
<i>ClusterRecOffset</i>	Aucun	0
<i>QType</i>	Aucun	0
<i>QDesc</i>	Aucun	" "
<i>DefBind</i>	Aucun	0
<i>DefPersistence</i>	Aucun	0
<i>DefPriority</i>	Aucun	0
<i>InhibitPut</i>	Aucun	0
<i>CLWLQueuePriority</i>	Aucun	0
<i>CLWLQueueRank</i>	Aucun	0
<i>DefPutResponse</i>	Aucun	1

**Remarques :**

1. Le symbole ↵ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable de macro MQWQR\_DEFAULT contient les valeurs par défaut. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:  

```
MQWQR MyWQR = {MQWQR_DEFAULT};
```
3. Les valeurs initiales définissent intentionnellement la longueur de la structure sur la longueur de la version en cours, et non sur la version 1 de la structure.

## Déclaration C

```
typedef struct tagMQWQR {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    StrucLength;      /* Length of MQWQR structure */
    MQLONG    QFlags;          /* Queue flags */
    MQCHAR48  QName;           /* Queue name */
};
```

```

MQCHAR48  QMgrIdentifier; /* Queue manageridentifier */
MQLONG   ClusterRecOffset; /* Offset of first cluster record */
MQLONG   QType; /* Queue type */
MQCHAR64  QDesc; /* Queue description */
MQLONG   DefBind; /* Default binding */
MQLONG   DefPersistence; /* Default message persistence */
MQLONG   DefPriority; /* Default message priority */
MQLONG   InhibitPut; /* Whether put operations on the queue
                        are allowed */

/* version 2 */
MQLONG   CLWLQueuePriority; /* Queue priority */
MQLONG   CLWLQueueRank; /* Queue rank */
/* version 3 */
MQLONG   DefPutResponse; /* Default put response */
};

```

## High Level Assembler

```

MQWQR                                DSECT
MQWQR_STRUCID                        DS    CL4    Structure identifier
MQWQR_VERSION                        DS    F      Structure version number
MQWQR_STRUCLength                    DS    F      Length of MQWQR structure
MQWQR_QFLAGS                         DS    F      Queue flags
MQWQR_QNAME                          DS    CL48   Queue name
MQWQR_QMGRIDENTIFIER                 DS    CL48   Queue manager identifier
MQWQR_CLUSTERRECOFFSET               DS    F      Offset of first cluster
*                                     record
MQWQR_QTYPE                          DS    F      Queue type
MQWQR_QDESC                          DS    CL64   Queue description
MQWQR_DEFBIND                        DS    F      Default binding
MQWQR_DEFPERSISTENCE                 DS    F      Default message persistence
MQWQR_DEFPRIORITY                    DS    F      Default message priority
MQWQR_INHIBITPUT                     DS    F      Whether put operations on
*                                     the queue are allowed
MQWQR_DEFPUTRESPONSE                 DS    F      Default put response
MQWQR_LENGTH                         EQU    *-MQWQR Length of structure
MQWQR_AREA                           ORG    MQWQR
MQWQR_AREA                           DS    CL(MQWQR_LENGTH)

```

### Référence associée

Zones dans [MQWQR](#) -Structure d'enregistrement de file d'attente de charge de travail de cluster  
Description des zones de la structure d'enregistrement de file d'attente de charge de travail de cluster MQWQR .

## MQWCR -Structure d'enregistrement de cluster de charge de travail de cluster

Le tableau suivant récapitule les zones de la structure d'enregistrement de charge de travail du cluster MQWCR .

Tableau 833. Zones dans MQWCR		
Zone	Description	Page
<i>ClusterName</i>	Nom du cluster	<a href="#">ClusterName</a>
<i>ClusterRecOffset</i>	Décalage de l'enregistrement de cluster suivant ( MQWCR )	<a href="#">ClusterRec</a>
<i>ClusterFlags</i>	Indicateurs de cluster	<a href="#">ClusterFlags</a>

La structure d'enregistrement de cluster de charge de travail de cluster contient des informations sur un cluster. Pour chaque cluster auquel appartient la file d'attente de destination, il existe une structure d'enregistrement de cluster de charge de travail de cluster.

La structure d'enregistrement de cluster de charge de travail de cluster est prise en charge dans tous les environnements.

### Référence associée

[MQ\\_CLUSTER\\_WORKLOAD\\_EXIT](#) -Description de l'appel

L'exit de charge de travail de cluster est appelé par le gestionnaire de files d'attente pour acheminer un message vers un gestionnaire de files d'attente disponible.

#### MQXCLWLN -Navigation dans les enregistrements de charge de travail de cluster

L'appel MQXCLWLN permet de naviguer dans les chaînes des enregistrements MQWDR, MQWQRet MQWCR stockés dans le cache du cluster.

#### MQWXP -Structure des paramètres d'exit de charge de travail de cluster

Le tableau suivant récapitule les zones de la structure de paramètres d'exit de charge de travail MQWXP -Cluster.

#### MQWDR-Structure d'enregistrement de destination de charge de travail de cluster

Le tableau suivant récapitule les zones de la structure d'enregistrement de destination de charge de travail MQWDR -Cluster.

#### MQWQR -Structure d'enregistrement de file d'attente de charge de travail de cluster

Le tableau suivant récapitule les zones de la structure d'enregistrement de file d'attente de charge de travail de cluster MQWQR .

### ***Zones de la structure d'enregistrement MQWCR -Cluster workload cluster.***

Description des zones de la structure d'enregistrement MQWCR -Cluster workload cluster.

#### **ClusterName ( MQCHAR48 ) -entrée**

Nom d'un cluster auquel appartient l'instance de la file d'attente de destination qui possède la structure MQWCR . L'instance de file d'attente de destination est décrite par une structure MQWDR .

- La longueur de ClusterName est MQ\_CLUSTER\_NAME\_LENGTH.

#### **ClusterRecClusterRec ( MQLONG ) -entrée**

Décalage logique de la structure MQWCR suivante.

- S'il n'existe plus de structures MQWCR , le décalage ClusterRec est égal à zéro.
- Le décalage est mesuré en octets à partir du début de la structure MQWCR .

#### **ClusterFlags ( MQLONG ) -entrée**

Les indicateurs de cluster indiquent les propriétés du gestionnaire de files d'attente identifié par la structure MQWCR . Les indicateurs suivants sont définis:

##### **MQQMF\_REPOSITORY\_Q\_MGR**

La destination est un gestionnaire de files d'attente de référentiel complet.

##### **MQQMF\_CLUSSDR\_USER\_DEFINED**

Le canal émetteur de cluster a été défini manuellement.

##### **MQQMF\_CLUSSDR\_AUTO\_DEFINED**

Le canal émetteur de cluster a été défini automatiquement.

##### **MQQMF\_AVAILABLE**

Le gestionnaire de files d'attente de destination est disponible pour recevoir des messages.

#### ***Autres valeurs***

D'autres indicateurs de la zone peuvent être définis par le gestionnaire de files d'attente à des fins internes.

#### **Référence associée**

Valeurs initiales et déclarations de langue pour MQWCR

Valeurs initiales et déclarations de langage C et High Level Assembler pour MQWCR -Structure d'enregistrement de cluster de charge de travail de cluster.

### ***Valeurs initiales et déclarations de langue pour MQWCR***

Valeurs initiales et déclarations de langage C et High Level Assembler pour MQWCR -Structure d'enregistrement de cluster de charge de travail de cluster.

Tableau 834. Zones dans MQWCR

Nom de zone	Nom de la constante	Valeur de la constante
<i>ClusterName</i>	Aucun	" "
<i>ClusterRecOffset</i>	Aucun	0
<i>ClusterFlags</i>	Aucun	0

## Déclaration C

```
typedef struct tagMQWCR {
    MQCHAR48 ClusterName; /* Cluster name */
    MQLONG ClusterRecOffset; /* Offset of next cluster record */
    MQLONG ClusterFlags; /* Cluster flags */
};
```

## High Level Assembler

```
MQWCR          DSECT
MQWCR_CLUSTERNAME DS CL48 Cluster name
MQWCR_CLUSTERRECOFFSET DS F Offset of next cluster
* record
MQWCR_CLUSTERFLAGS DS F Cluster flags
MQWCR_LENGTH EQU *-MQWCR Length of structure
MQWCR_ORG ORG MQWCR
MQWCR_AREA DS CL(MQWCR_LENGTH)
```

## Référence associée

[Zones de la structure d'enregistrement MQWCR -Cluster workload cluster.](#)

Description des zones de la structure d'enregistrement MQWCR -Cluster workload cluster.

## Référence d'exit API

Cette section fournit des informations de référence présentant principalement un intérêt pour un programmeur qui écrit des exits API.

## Remarques générales sur l'utilisation

### de document :

1. Toutes les fonctions d'exit peuvent émettre l'appel MQXEP ; cet appel est conçu spécifiquement pour être utilisé à partir des fonctions d'exit d'API.
2. La fonction MQ\_INIT\_EXIT ne peut pas émettre d'appels MQ autres que MQXEP.
3. Vous ne pouvez pas émettre l'appel MQDISC pour la connexion en cours.
4. Si une fonction d'exit émet l'appel MQCONN ou l'appel MQCONNX avec l'option MQCNO\_HANDLE\_SHARE\_NONE, l'appel se termine avec le code anomalie MQRC\_ALREADY\_CONNECTED et le descripteur renvoyé est identique à celui transmis à l'exit en tant que paramètre.
5. En général, lorsqu'une fonction d'exit API émet un appel MQI, les exits API ne sont pas appelés de manière récursive. Toutefois, si une fonction d'exit émet l'appel MQCONNX avec les options MQCNO\_HANDLE\_SHARE\_BLOCK ou MQCNO\_HANDLE\_SHARE\_NO\_BLOCK, l'appel renvoie un nouveau descripteur partagé. Cela fournit à la suite d'exit un descripteur de connexion qui lui est propre et donc une unité de travail indépendante de l'unité de travail de l'application. La suite de sorties peut utiliser cet indicateur pour insérer et extraire des messages dans sa propre unité de travail, et valider ou abandonner cette unité de travail ; tout cela peut être fait sans affecter de quelque manière que ce soit l'unité de travail de l'application.

Etant donné que la fonction d'exit utilise un descripteur de connexion différent de celui utilisé par l'application, les appels MQ émis par la fonction d'exit entraînent l'appel des fonctions d'exit d'API appropriées. Les fonctions de sortie peuvent donc être appelées de manière récursive. Notez que la zone *ExitUserArea* dans MQAXP et la zone de chaîne d'exit ont une portée de descripteur de connexion. Par conséquent, une fonction d'exit ne peut pas utiliser ces zones pour signaler à une autre instance qu'elle a appelée de manière récursive qu'elle est déjà active.

6. Les fonctions de sortie peuvent également insérer et extraire des messages dans l'unité d'oeuvre de l'application. Lorsque l'application valide ou annule l'unité de travail, tous les messages de l'unité de travail sont validés ou annulés ensemble, quelle que soit la personne qui les a placés dans l'unité de travail (application ou fonction de sortie). Cependant, l'exit peut amener l'application à dépasser les limites du système plus tôt que dans le cas contraire (par exemple, en dépassant le nombre maximal de messages non validés dans une unité de travail).

Lorsqu'une fonction d'exit utilise l'unité de travail de l'application de cette manière, la fonction d'exit doit généralement éviter d'émettre l'appel MQCMIT, car elle valide l'unité de travail de l'application et peut nuire au bon fonctionnement de l'application. Toutefois, la fonction d'exit peut parfois avoir besoin d'émettre l'appel MQBACK si la fonction d'exit rencontre une erreur grave qui empêche la validation de l'unité de travail (par exemple, une erreur lors de l'insertion d'un message dans l'unité de travail de l'application). Lorsque MQBACK est appelé, veillez à ce que les limites de l'unité de travail de l'application ne soient pas modifiées. Dans ce cas, la fonction d'exit doit définir les valeurs appropriées pour s'assurer que le code achèvement MQCC\_WARNING et le code anomalie MQRC\_BACKED\_OUT sont renvoyés à l'application, afin que l'application puisse détecter le fait que l'unité de travail a été annulée.

Si une fonction d'exit utilise le descripteur de connexion de l'application pour émettre des appels MQ, ces appels ne génèrent pas eux-mêmes d'autres appels de fonctions d'exit d'API.

7. Si une fonction d'exit MQXR\_BEFORE s'arrête de manière anormale, le gestionnaire de files d'attente peut être en mesure d'effectuer une reprise après incident. Si tel est le cas, le gestionnaire de files d'attente poursuit le traitement comme si la fonction d'exit avait renvoyé MQXCC\_FAILED. Si le gestionnaire de files d'attente ne peut pas effectuer de reprise, l'application est arrêtée.
8. Si une fonction d'exit MQXR\_AFTER s'arrête de manière anormale, le gestionnaire de files d'attente peut être en mesure d'effectuer une reprise après incident. Si tel est le cas, le gestionnaire de files d'attente poursuit le traitement comme si la fonction d'exit avait renvoyé MQXCC\_FAILED. Si le gestionnaire de files d'attente ne peut pas effectuer de reprise, l'application est arrêtée. Sachez que dans ce dernier cas, les messages extraits en dehors d'une unité de travail sont perdus (c'est la même situation que l'application qui échoue immédiatement après la suppression d'un message de la file d'attente).
9. Le processus MCA effectue une validation en deux phases.

Si un exit API intercepte un MQCMIT d'un processus MCA préparé et tente d'effectuer une action dans l'unité de travail, l'action échoue avec le code anomalie MQRC\_UOW\_NOT\_AVAILABLE.

10. Lorsque plusieurs installations de IBM MQ sont disponibles, utilisez les exits écrits pour une version antérieure de IBM MQ, car les nouvelles fonctionnalités ajoutées dans la version ultérieure risquent de ne pas fonctionner avec les versions antérieures. Pour plus d'informations sur les modifications entre les éditions, voir [Nouveautés dans IBM MQ 8.0](#).

## Structure des paramètres d'exit API IBM MQ (MQAXP)

La structure MQAXP, un bloc de contrôle externe, est utilisée comme paramètre d'entrée ou de sortie de l'exit API. Cette rubrique fournit également des informations sur le fonctionnement de l'exit de processus des gestionnaires de files d'attente.

MQAXP possède la déclaration C suivante:

```
typedef struct tagMQAXP {
    MQCHAR4   StructId;           /* Structure identifier */
    MQLONG    Version;           /* Structure version number */
    MQLONG    ExitId;            /* Exit Identifier */
    MQLONG    ExitReason;        /* Exit invocation reason */
}
```

```

MQLONG    ExitResponse;        /* Response code from exit */
MQLONG    ExitResponse2;      /* Secondary response code from exit */
MQLONG    Feedback;           /* Feedback code from exit */
MQLONG    APICallerType;      /* MQSeries API caller type */
MQBYTE16  ExitUserArea;       /* User area for use by exit */
MQCHAR32  ExitData;           /* Exit data area */
MQCHAR48  ExitInfoName;       /* Exit information name */
MQBYTE48  ExitPDArea;         /* Problem determination area */
MQCHAR48  QMgrName;           /* Name of local queue manager */
PMQACH    ExitChainAreaPtr;   /* Inter exit communication area */
MQHCONFIG  Hconfig;           /* Configuration handle */
MQLONG    Function;           /* Function Identifier */
/* Ver:1 */
MQHMSG    ExitMsgHandle        /* Exit message handle
/* Ver:2 */
};

```

La liste de paramètres suivante est transmise lorsque des fonctions d'un exit API sont appelées:

#### **StrucId (MQCHAR4)-entrée**

Identificateur de la structure du paramètre d'exit, avec la valeur:

```
MQAXP_STRUC_ID.
```

Le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit.

#### **Version (MQLONG)-entrée**

Numéro de version de la structure, avec la valeur:

##### **MQAXP\_VERSION\_1**

Structure des paramètres d'exit d'API version 1.

##### **MQAXP\_VERSION\_2**

Structure des paramètres d'exit d'API version 2.

##### **MQAXP\_CURRENT\_VERSION**

Numéro de version en cours de la structure de paramètres d'exit d'API.

Le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit.

#### **ExitId (MQLONG)-entrée**

Identificateur de l'exit, défini à l'entrée de la routine d'exit, indiquant le type de l'exit:

##### **MQXT\_API\_EXIT**

Exit API.

#### **ExitReason (MQLONG)-entrée**

Motif de l'appel de l'exit, défini à l'entrée de chaque fonction d'exit:

##### **MQXR\_CONNECTION**

L'exit est appelé pour s'initialiser avant un appel MQCONN ou MQCONNX ou pour se terminer après un appel MQDISC.

##### **MQXR\_AVANT**

L'exit est appelé avant l'exécution d'un appel API ou avant la conversion de données sur une requête MQGET.

##### **MQXR\_APRES**

L'exit est appelé après l'exécution d'un appel API.

#### **ExitResponse (MQLONG)-sortie**

La réponse de l'exit, initialisée à l'entrée de chaque fonction d'exit pour:

##### **MQXCC\_OK**

Continuez normalement.

Cette zone doit être définie par la fonction d'exit pour communiquer au gestionnaire de files d'attente le résultat de l'exécution de la fonction d'exit. Il doit s'agir de l'une des valeurs suivantes :

##### **MQXCC\_OK**

La fonction d'exit a abouti. Continuez normalement.



Cette valeur peut être définie par toutes les fonctions d'exit MQXR\_\*. ExitResponse2 est utilisé pour décider si les fonctions de sortie doivent être appelées ultérieurement dans la chaîne.

### **MQXCC\_ECHEC**

La fonction d'exit a échoué en raison d'une erreur.

Cette valeur peut être définie par toutes les fonctions d'exit MQXR\_\*. Le gestionnaire de files d'attente définit CompCode sur MQCC\_FAILED et la raison sur:

- MQRC\_API\_EXIT\_INIT\_ERROR si la fonction est MQ\_INIT\_EXIT
- MQRC\_API\_EXIT\_TERM\_ERROR si la fonction est MQ\_TERM\_EXIT
- MQRC\_API\_EXIT\_ERROR pour toutes les autres fonctions d'exit

Les valeurs définies peuvent être modifiées par une fonction de sortie plus tard dans la chaîne.

ExitResponse2 est ignoré ; le gestionnaire de files d'attente poursuit le traitement comme si MQXR2\_SUPPRESS\_CHAIN avait été renvoyé.

### **MQXCC\_SUPPRESS\_FUNCTION**

Supprimez la fonction d'API IBM MQ.

Cette valeur ne peut être définie que par une fonction de sortie MQXR\_BEFORE. Il ignore l'appel d'API. S'il est renvoyé par MQ\_DATA\_CONV\_ON\_GET\_EXIT, la conversion de données est ignorée. Le gestionnaire de files d'attente définit CompCode sur MQCC\_FAILED et Cause sur MQRC\_SUPPRESSED\_BY\_EXIT, mais les valeurs définies peuvent être modifiées par une fonction d'exit plus loin dans la chaîne. Les autres paramètres de l'appel restent tels que l'exit les a laissés. ExitResponse2 est utilisé pour décider si les fonctions de sortie doivent être appelées ultérieurement dans la chaîne.

Si cette valeur est définie par une fonction d'exit MQXR\_AFTER ou MQXR\_CONNECTION, le gestionnaire de files d'attente poursuit le traitement comme si MQXCC\_FAILED avait été renvoyé.

### **MQXCC\_LISTE\_FONCTION**

Ignorez la fonction d'API IBM MQ.

Cette valeur ne peut être définie que par une fonction de sortie MQXR\_BEFORE. Il ignore l'appel d'API. S'il est renvoyé par MQ\_DATA\_CONV\_ON\_GET\_EXIT, la conversion de données est ignorée. La fonction d'exit doit définir CompCode et Raison sur les valeurs à renvoyer à l'application, mais les valeurs définies peuvent être modifiées par une fonction d'exit plus tard dans la chaîne. Les autres paramètres de l'appel restent tels que l'exit les a laissés. ExitResponse2 est utilisé pour décider si les fonctions de sortie doivent être appelées ultérieurement dans la chaîne.

Si cette valeur est définie par une fonction d'exit MQXR\_AFTER ou MQXR\_CONNECTION, le gestionnaire de files d'attente poursuit le traitement comme si MQXCC\_FAILED avait été renvoyé.

### **MQXCC\_SUPPRESS\_EXIT**

Supprimez toutes les fonctions d'exit appartenant à l'ensemble d'exits.

Cette valeur ne peut être définie que par les fonctions d'exit MQXR\_BEFORE et MQXR\_AFTER. Il ignore *tous* les appels ultérieurs des fonctions d'exit appartenant à cet ensemble d'exits pour cette connexion logique. Ce contournement se poursuit jusqu'à ce que la demande de déconnexion logique se produise, lorsque la fonction MQ\_TERM\_EXIT est appelée avec un ExitReason de MQXR\_CONNECTION.

La fonction d'exit doit définir CompCode et Raison sur les valeurs à renvoyer à l'application, mais les valeurs définies peuvent être modifiées par une fonction d'exit plus tard dans la chaîne. Les autres paramètres de l'appel restent tels que l'exit les a laissés. ExitResponse2 est ignoré.

Si cette valeur est définie par une fonction d'exit MQXR\_CONNECTION, le gestionnaire de files d'attente poursuit le traitement comme si MQXCC\_FAILED avait été renvoyé.

Pour plus d'informations sur l'interaction entre ExitResponse et ExitResponse2 et sur son effet sur le traitement de sortie, voir [«Fonctionnement de l'exit de processus des gestionnaires de files d'attente»](#), à la page 1644.

## ExitResponse2 (MQLONG)-sortie

Il s'agit d'un code de réponse d'exit secondaire qui qualifie le code de réponse d'exit principal pour les fonctions d'exit MQXR\_BEFORE. Il est initialisé pour:

```
MQXR2_DEFAULT_CONTINUATION
```

sur l'entrée d'une fonction d'exit d'appel d'API IBM MQ . Il peut ensuite être défini sur l'une des valeurs suivantes:

### MQXR2\_DEFAULT\_CONTINUATION

Indique s'il faut continuer avec l'exit suivant dans la chaîne, en fonction de la valeur de ExitResponse.

Si ExitResponse est MQXCC\_SUPPRESS\_FUNCTION ou MQXCC\_SKIP\_FUNCTION, ignorez les fonctions d'exit ultérieurement dans la chaîne MQXR\_BEFORE et les fonctions d'exit correspondantes dans la chaîne MQXR\_AFTER. Appelez les fonctions d'exit dans la chaîne MQXR\_AFTER qui correspondent aux fonctions d'exit précédemment dans la chaîne MQXR\_BEFORE.

Sinon, appelez l'exit suivant dans la chaîne.

### MQXR2\_SUPPRESS\_CHAIN

Supprimez la chaîne.

Ignorez les fonctions d'exit ultérieurement dans la chaîne MQXR\_BEFORE et les fonctions d'exit correspondantes dans la chaîne MQXR\_AFTER pour cet appel d'API. Appelez les fonctions d'exit dans la chaîne MQXR\_AFTER qui correspondent aux fonctions d'exit précédemment dans la chaîne MQXR\_BEFORE.

### MQXR2\_CONTINUE\_CHAIN

Passez à l'exit suivant dans la chaîne.

Pour plus d'informations sur l'interaction entre ExitResponse et ExitResponse2 et sur son effet sur le traitement de sortie, voir «[Fonctionnement de l'exit de processus des gestionnaires de files d'attente](#)», à la page 1644.

## Commentaires en retour (MQLONG)-entrée / sortie

Communiquez les codes retour entre les appels de fonction de sortie. Cette opération est initialisée pour:

```
MQFB_NONE (0)
```

avant d'invoquer la première fonction de la première sortie dans une chaîne.

Les exits peuvent définir cette zone sur n'importe quelle valeur, y compris toute valeur MQFB\_\* ou MQRC\_\* valide. Les exits peuvent également définir cette zone sur une valeur de retour d'informations définie par l'utilisateur comprise entre MQFB\_APPL\_FIRST et MQFB\_APPL\_LAST.

## APICallerType (MQLONG)-entrée

Type d'appelant de l'API, indiquant si l'appelant de l'API IBM MQ est externe ou interne au gestionnaire de files d'attente: MQXACT\_EXTERNAL ou MQXACT\_INTERNAL.

## Zone ExitUser(MQBYTE16)-entrée / sortie

Zone utilisateur, disponible pour tous les exits associés à un objet ExitInfoparticulier. Il est initialisé sur MQXUA\_NONE (zéros binaires pour la longueur de la zone ExitUser) avant d'appeler la première fonction d'exit (MQ\_INIT\_EXIT) pour hconn. A partir de ce moment, toutes les modifications apportées à cette zone par une fonction d'exit sont conservées dans les appels de fonctions du même exit.

Cette zone est alignée sur un multiple de 4 MQLONGs.

Les exits peuvent également ancrer tout stockage qu'ils allouent à partir de cette zone.

Pour chaque hconn, chaque exit d'une chaîne d'exits possède une zone ExitUserdifférente. La zone ExitUserne peut pas être partagée par les exits d'une chaîne et le contenu de la zone ExitUserd'un exit n'est pas disponible pour un autre exit d'une chaîne.

Pour les programmes C, la constante MQXUA\_NONE\_ARRAY est également définie avec la même valeur que MQXUA\_NONE, mais sous la forme d'un tableau de caractères au lieu d'une chaîne.

La longueur de cette zone est indiquée par MQ\_EXIT\_USER\_AREA\_LENGTH.

#### **ExitData (MQCHAR32)-entrée**

Données d'exit, définies à l'entrée de chaque fonction d'exit sur les 32 caractères des données spécifiques à l'exit qui sont fournies dans l'exit. Si vous ne définissez aucune valeur dans l'exit, cette zone est vide.

La longueur de cette zone est donnée par MQ\_EXIT\_DATA\_LENGTH.

#### **ExitInfoNom (MQCHAR48)-entrée**

Nom des informations d'exit, défini à l'entrée de chaque fonction d'exit sur le nom ApiExit\_name spécifié dans les définitions d'exit dans les sections.

#### **ExitPDArea (MQBYTE48)-entrée / sortie**

Une zone d'identification des incidents, initialisée à MQXPDA\_NONE (zéros binaires pour la longueur de la zone) pour chaque appel d'une fonction d'exit.

Pour les programmes C, la constante MQXPDA\_NONE\_ARRAY est également définie avec la même valeur que MQXPDA\_NONE, mais sous la forme d'un tableau de caractères au lieu d'une chaîne.

Le gestionnaire d'exit écrit toujours cette zone dans la trace IBM MQ à la fin d'un exit, même lorsque la fonction aboutit.

La longueur de cette zone est indiquée par MQ\_EXIT\_PD\_AREA\_LENGTH.

#### **QMgrName (MQCHAR48)-entrée**

Nom du gestionnaire de files d'attente auquel l'application est connectée, qui a appelé un exit suite au traitement d'un appel d'API IBM MQ .

Si le nom d'un gestionnaire de files d'attente fourni sur un appel MQCONN ou MQCONNX est vide, cette zone est toujours définie sur le nom du gestionnaire de files d'attente auquel l'application est connectée, que l'application soit serveur ou client.

Le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit.

La longueur de cette zone est indiquée par MQ\_Q\_MGR\_NAME\_LENGTH.

#### **ExitChainAreaPtr (PMQACH)-entrée/sortie**

Permet de communiquer des données entre les appels de différents exits dans une chaîne. Il est défini sur un pointeur NULL avant d'appeler la première fonction (MQ\_INIT\_EXIT avec ExitReason MQXR\_CONNECTION) du premier exit dans une chaîne d'exits. La valeur renvoyée par l'exit lors d'un appel est transmise à l'appel suivant.

Pour plus d'informations sur l'utilisation de la zone de chaîne d'exit, voir [«Zone de chaîne d'exit et en-tête de zone de chaîne d'exit \(MQACH\)»](#), à la page 1648 .

#### **Hconfig (MQHCONFIG)-entrée**

Descripteur de configuration représentant l'ensemble des fonctions en cours d'initialisation. Cette valeur est générée par le gestionnaire de files d'attente sur la fonction MQ\_INIT\_EXIT et est ensuite transmise à la fonction d'exit d'API. Il est défini à l'entrée de chaque fonction d'exit.

Vous pouvez utiliser Hconfig comme pointeur vers la structure MQIEP pour effectuer des appels MQI et DCI. Vous devez vérifier que les 4 premiers octets de HConfig correspondent à l' StrucId de la structure MQIEP avant d'utiliser le paramètre HConfig comme pointeur vers la structure MQIEP.

#### **Fonction (MQLONG)-entrée**

Identificateur de la fonction, dont les valeurs valides sont les constantes MQXF\_ \* décrites dans [«Constantes externes»](#), à la page 1649.

Le gestionnaire d'exit définit cette zone sur la valeur correcte, à l'entrée de chaque fonction d'exit, en fonction de l'appel de l'API IBM MQ qui a entraîné l'appel de l'exit.

### ExitMsgIdentificateur (MQHMSG)-entrée/sortie

Lorsque la fonction est MQXF\_GET et que ExitReason est MQXR\_AFTER, un descripteur de message valide est renvoyé dans cette zone pour permettre à l'exit d'API d'accéder aux zones de descripteur de message et à toute autre propriété correspondant à la chaîne ExitProperties spécifiée dans la structure MQXEPO lors de l'enregistrement de l'exit d'API.

Toutes les propriétés de descripteur de message qui sont renvoyées dans le descripteur ExitMsgne seront pas disponibles à partir de MsgHandle dans la structure MQGMO, si elle a été spécifiée, ou dans les données de message.

Lorsque la fonction est MQXF\_GET et que ExitReason est MQXR\_BEFORE, si le programme d'exit définit cette zone sur MQHM\_NONE, il supprime le remplissage des propriétés de descripteur ExitMsg.

Cette zone n'est pas définie si la version est inférieure à MQAXP\_VERSION\_2.

### Fonctionnement de l'exit de processus des gestionnaires de files d'attente

Le traitement effectué par le gestionnaire de files d'attente en cas de retour d'une fonction d'exit dépend à la fois de ExitResponse et de ExitResponse2.

Le [Tableau 835](#), à la page 1644 récapitule les combinaisons possibles et leurs effets pour une fonction d'exit MQXR\_BEFORE, en indiquant:

- Qui définit les paramètres CompCode et Reason de l'appel API
- Indique si les fonctions d'exit restantes dans la chaîne MQXR\_BEFORE et les fonctions d'exit correspondantes dans la chaîne MQXR\_AFTER sont appelées
- Indique si l'appel d'API est appelé

Pour une fonction d'exit MQXR\_AFTER:

- CompCode et Reason sont définis de la même manière que MQXR\_BEFORE
- ExitResponse2 est ignoré (les autres fonctions d'exit de la chaîne MQXR\_AFTER sont toujours appelées)
- MQXCC\_SUPPRESS\_FUNCTION et MQXCC\_SKIP\_FUNCTION ne sont pas valides

Pour une fonction d'exit MQXR\_CONNECTION:

- CompCode et Reason sont définis de la même manière que MQXR\_BEFORE
- ExitResponse2 est ignoré
- MQXCC\_SUPPRESS\_FUNCTION, MQXCC\_SKIP\_FUNCTION, MQXCC\_SUPPRESS\_EXIT ne sont pas valides

Dans tous les cas où un exit ou le gestionnaire de files d'attente définit CompCode et Reason, les valeurs définies peuvent être modifiées par un exit appelé ultérieurement ou par l'appel API (si l'appel API est appelé ultérieurement).

<i>Tableau 835. Traitement de l'exit MQXR_BEFORE</i>			
<b>Valeur de ExitResponse</b>	<b>CompCode et raison définie par</b>	<b>Valeur de ExitResponse2 (continuation par défaut) Chain</b>	<b>Valeur de l'API ExitResponse2 (continuation par défaut)</b>
MQXCC_OK	quitter	Y	Y
MQXCC_SUPPRESS_EXIT	quitter	Y	Y
MQXCC_SUPPRESS_FUNCTION	gestionnaire de files d'attente	N	N

Tableau 835. Traitement de l'exit MQXR\_BEFORE (suite)

Valeur de ExitResponse	CompCode et raison définie par	Valeur de ExitResponse2 (continuation par défaut) Chain	Valeur de l'API ExitResponse2 (continuation par défaut)
FONCTION MQXCC_SKIP	quitter	N	N
MQXCC_ECHEC	gestionnaire de files d'attente	N	N

## Fonctionnement de l'exit de processus des clients

En général, les clients traitent les fonctions d'exit de la même manière que les applications serveur et l'attribut *QMgrName* de cette structure s'applique que la fonction se trouve sur un serveur ou sur un client.

Toutefois, le client n'ayant aucun concept du fichier *mqs.ini*, les sections *ApiExitCommon* et *APIExitTemplate* ne s'appliquent pas. Seule la section *ApiExitLocal* s'applique et cette section est configurée dans le fichier *mqclient.ini*.

## Structure de contexte d'exit API IBM MQ (MQAXC)

La structure MQAXC, un bloc de contrôle externe, est utilisée comme paramètre d'entrée pour un exit d'API.

MQAXC possède la déclaration C suivante:

```
typedef struct tagMQAXC {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    Environment;      /* Environment */
    MQCHAR12  UserId;           /* UserId associated with appl */
    MQBYTE40  SecurityId        /* Extension to UserId running appl */
    MQCHAR264 ConnectionName;    /* Connection name */
    MQLONG    LongMCAUserIdLength; /* long MCA user identifier length */
    MQLONG    LongRemoteUserIdLength; /* long remote user identifier length */
    MQPTR     LongMCAUserIdPtr;  /* long MCA user identifier address */
    MQPTR     LongRemoteUserIdPtr; /* long remote user identifier address */
    MQCHAR28  ApplName;         /* Application name */
    MQLONG    ApplType;         /* Application type */
    MQPID     ProcessId;        /* Process identifier */
    MQTID     ThreadId;         /* Thread identifier */

    /* Ver:1 */
    MQCHAR    ChannelName[20]    /* Channel Name */
    MQBYTE4   Reserved1;        /* Reserved */
    PMQCD     pChannelDefinition; /* Channel Definition pointer */
};
```

Les paramètres de MQAXC sont les suivants:

### StrucId (MQCHAR4)-entrée

Identificateur de la structure de contexte d'exit, avec la valeur MQAXC\_STRUC\_ID. Pour les programmes C, la constante MQAXC\_STRUC\_ID\_ARRAY est également définie, avec la même valeur que MQAXC\_STRUC\_ID, mais sous la forme d'un tableau de caractères au lieu d'une chaîne.

Le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit.

### Version (MQLONG)-entrée

Numéro de version de la structure, avec la valeur:

#### MQAXC\_VERSION\_2

Numéro de version de la structure de contexte d'exit.

#### MQAXC\_CURRENT\_VERSION

Numéro de version en cours de la structure de contexte d'exit.

Le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit.

### **Environnement (MQLONG)-entrée**

Environnement à partir duquel un appel d'API IBM MQ a été émis qui a entraîné l'exécution d'une fonction d'exit. Les valeurs admises pour cette zone sont les suivantes:

#### **MQXE\_AUTRES**

Cette valeur est cohérente avec les appels qu'un exit API voit si l'exit est appelé à partir d'une application serveur. Cela signifie qu'un exit d'API s'exécute sans modification sur un client et ne voit rien de différent.

Si l'exit a réellement besoin de déterminer s'il est en cours d'exécution sur le client, il peut le faire en consultant les zones *ChannelName* et *ChannelDefinition* .

#### **MQXE\_MCA**

Agent MCA

#### **MQXE\_MCA\_SVRCONN**

Un agent MCA agissant pour le compte d'un client

#### **SERVEUR\_COMMAND\_MQ**

Serveur de commandes

#### **MQXE\_MQSC**

Interpréteur de commandes runmqsc

Le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit.

### **UserId (MQCHAR12)-entrée**

ID utilisateur associé à l'application. En particulier, dans le cas de connexions client, cette zone contient l'ID utilisateur de l'utilisateur adopté par opposition à l'ID utilisateur sous lequel le code de canal est exécuté. Si un ID utilisateur vide provient du client, aucune modification n'est apportée à l'ID utilisateur déjà utilisé. Autrement dit, aucun nouvel ID utilisateur n'est adopté.

Le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit. La longueur de cette zone est indiquée par MQ\_USER\_ID\_LENGTH.

Dans le cas d'un client, il s'agit de l'ID utilisateur envoyé du client au serveur. Notez qu'il peut ne pas s'agir de l'ID utilisateur effectif sur lequel le client s'exécute dans le gestionnaire de files d'attente, car il peut y avoir une configuration MCAUser ou CHLAUTH qui modifie l'ID utilisateur.

### **SecurityId (MQBYTE40)-entrée**

Extension de l'ID utilisateur exécutant l'application. Sa longueur est donnée par MQ\_SECURITY\_ID\_LENGTH.

Dans le cas d'un client, il s'agit de l'ID utilisateur envoyé du client au serveur. Notez qu'il peut ne pas s'agir de l'ID utilisateur effectif sur lequel le client s'exécute dans le gestionnaire de files d'attente, car il peut y avoir une configuration MCAUser ou CHLAUTH qui modifie l'ID utilisateur.

### **ConnectionName (MQCHAR264)-entrée**

Zone de nom de connexion, définie sur l'adresse du client. Par exemple, pour TCP/IP, il s'agit de l'adresse IP du client.

La longueur de cette zone est indiquée par MQ\_CONN\_NAME\_LENGTH.

Dans le cas d'un client, il s'agit de l'adresse du partenaire du gestionnaire de files d'attente.

### **LongMCAUserIdLength (MQLONG)-entrée**

Longueur de l'ID utilisateur MCA long.

Lorsque MCA se connecte au gestionnaire de files d'attente, cette zone est définie sur la longueur de l'ID utilisateur MCA long (ou sur zéro s'il n'existe pas d'identificateur de ce type).

Dans le cas d'un client, il s'agit de l'identificateur d'utilisateur long client.

### **LongRemoteUserIdLongueur (MQLONG)-entrée**

Longueur de l'ID utilisateur éloigné long.

Lorsque MCA se connecte au gestionnaire de files d'attente, cette zone est définie sur la longueur de l'ID utilisateur distant long. Sinon, cette zone sera définie sur zéro

Dans le cas d'un client, définissez cette zone sur zéro.

#### **LongMCAUserIdPtr (MQPTR)-entrée**

Adresse de l'ID utilisateur MCA long.

Lorsque MCA se connecte au gestionnaire de files d'attente, cette zone est définie sur l'adresse de l'identificateur utilisateur MCA long (ou sur un pointeur NULL s'il n'existe pas d'identificateur de ce type).

Dans le cas d'un client, il s'agit de l'identificateur d'utilisateur long client.

#### **LongRemoteUserIdPtr (MQPTR)-entrée**

Adresse de l'ID utilisateur distant long.

Lorsque MCA se connecte au gestionnaire de files d'attente, cette zone est définie sur l'adresse de l'identificateur d'utilisateur distant long (ou sur un pointeur NULL s'il n'existe pas d'identificateur de ce type).

Dans le cas d'un client, définissez cette zone sur zéro.

#### **ApplName (MQCHAR28)-entrée**

Nom de l'application ou du composant qui a émis l'appel d'API IBM MQ .

Les règles de génération du ApplName sont les mêmes que pour la génération du nom par défaut d'une instruction MQPUT.

La valeur de cette zone est trouvée en demandant le nom du programme au système d'exploitation. Sa longueur est donnée par MQ\_APPL\_NAME\_LENGTH.

#### **ApplType (MQLONG)-entrée**

Type d'application ou de composant ayant émis l'appel d'API IBM MQ .

La valeur est MQAT\_DEFAULT pour la plateforme sur laquelle l'application est compilée ou correspond à l'une des valeurs MQAT\_\* définies.

Le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit.

#### **ProcessId (MQPID)-entrée**

Identificateur de processus du système d'exploitation.

Le cas échéant, le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit.

#### **ThreadId (MQTID)-entrée**

Identificateur de l'unité d'exécution MQ . Il s'agit du même identificateur utilisé dans la trace MQ et les vidages FFST , mais il peut être différent de l'identificateur d'unité d'exécution du système d'exploitation.

Le cas échéant, le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit.

#### **ChannelName (MQCHAR)-entrée**

Nom du canal, complété par des blancs, le cas échéant et connu.

S'il n'est pas applicable, cette zone est définie sur des caractères NULL.

#### **Reserved1 (MQBYTE4)-entrée**

Cette zone est réservée.

#### **ChanneDefinition (PMQCD)-entrée**

Un pointeur vers la définition de canal utilisée, le cas échéant et connue.

S'il n'est pas applicable, cette zone est définie sur des caractères NULL.

Notez que le pointeur n'est terminé que si la connexion est en cours de traitement pour le compte d'un canal IBM MQ et que cette définition de canal a été lue.

En particulier, la définition de canal n'est pas donnée sur le serveur lorsque le premier appel MQCONN est effectué pour le canal. En outre, si le pointeur est rempli, la structure (et les éventuelles sous-

structures) pointée par le pointeur doivent être traitées comme étant en lecture seule ; toute mise à jour de la structure conduirait à des résultats imprévisibles et n'est pas supportée.

Dans le cas d'un client, les zones autres que celles dont la valeur est spécifiée pour un client contiennent des valeurs appropriées pour une application client.

## Zone de chaîne d'exit et en-tête de zone de chaîne d'exit (MQACH)

Si nécessaire, une fonction d'exit peut acquérir du stockage pour une zone de chaîne d'exit et définir le paramètre ExitChainAreaPtr dans MQAXP pour qu'il pointe vers ce stockage.

Les exits (des fonctions d'exit identiques ou différentes) peuvent acquérir plusieurs zones de chaîne d'exit et les lier ensemble. Les zones de chaîne d'exit doivent uniquement être ajoutées ou supprimées de cette liste lorsqu'elles sont appelées à partir du gestionnaire d'exit. Cela permet de s'assurer qu'il n'y a pas de problèmes de sérialisation causés par des unités d'exécution différentes qui ajoutent ou suppriment des zones de la liste en même temps.

Une zone de chaîne d'exit doit commencer par une structure d'en-tête MQACH, dont la déclaration C est:

```
typedef struct tagMQACH {
    MQCHAR4   StructId;           /* Structure identifier */
    MQLONG    Version;           /* Structure version number */
    MQLONG    StructLength;      /* Length of the MQACH structure */
    MQLONG    ChainAreaLength;   /* Exit chain area length */
    MQCHAR48  ExitInfoName;     /* Exit information name */
    PMQACH    NextChainAreaPtr; /* Pointer to next exit chain area */
};
```

Les zones de l'en-tête de la zone de chaîne d'exit sont les suivantes:

### StrucId (MQCHAR4)-entrée

Identificateur de structure de zone de chaîne d'exit, avec une valeur initiale, définie par MQACH\_DEFAULT, de MQACH\_STRUC\_ID.

Pour les programmes C, la constante MQACH\_STRUC\_ID\_ARRAY est également définie ; elle a la même valeur que MQACH\_STRUC\_ID, mais sous la forme d'un tableau de caractères au lieu d'une chaîne.

### Version (MQLONG)-entrée

Numéro de version de la structure, comme suit:

#### MQACH\_VERSION\_1

Numéro de version de la structure du paramètre d'exit.

#### VERSION\_MQACH\_CURRENT\_VERSION

Numéro de version en cours de la structure de contexte d'exit.

La valeur initiale de cette zone, définie par MQACH\_DEFAULT, est MQACH\_CURRENT\_VERSION.

**Remarque :** Si vous introduisez une nouvelle version de cette structure, la présentation de la partie existante ne change pas. Les fonctions d'exit doivent vérifier que le numéro de version est égal ou supérieur à la version la plus basse contenant les zones que la fonction d'exit doit utiliser.

### StructLength (MQLONG)-entrée

Longueur de la structure MQACH. Les exits peuvent utiliser cette zone pour déterminer le début des données d'exit, en lui affectant la longueur de la structure créée par l'exit.

La valeur initiale de cette zone, définie par MQACH\_DEFAULT, est MQACH\_CURRENT\_LENGTH.

### ChainAreaLongueur (MQLONG)-entrée

Longueur de la zone de chaîne d'exit, définie sur la longueur totale de la zone de chaîne d'exit en cours, y compris l'en-tête MQACH.

La valeur initiale de cette zone, définie par MQACH\_DEFAULT, est zéro.

### ExitInfoNom (MQCHAR48)-entrée

Nom des informations d'exit.



Lorsqu'un exit crée une structure MQACH, il doit initialiser cette zone avec son propre nom ExitInfo, de sorte que cette structure MQACH puisse être trouvée ultérieurement par une autre instance de cet exit ou par un exit associé.

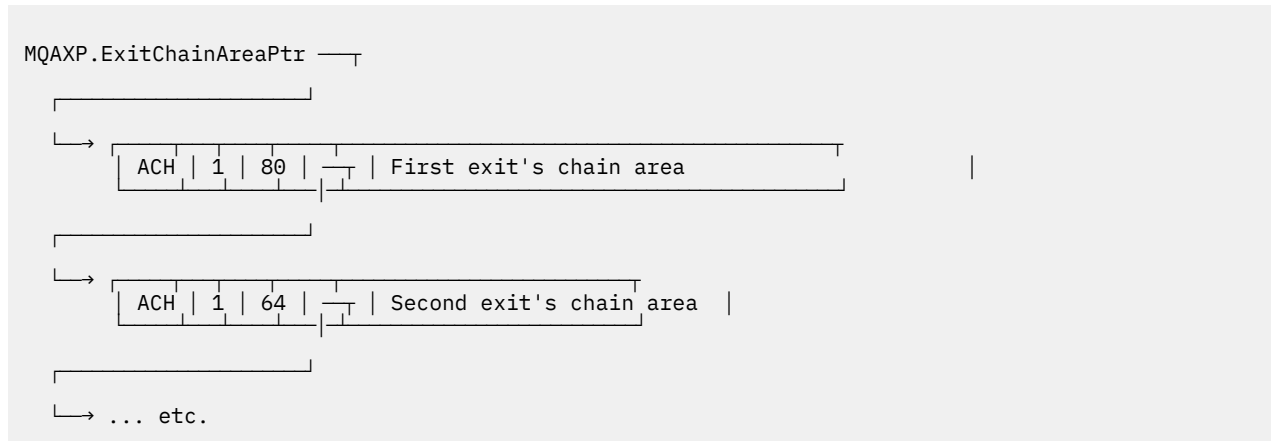
La valeur initiale de cette zone, définie par MQACH\_DEFAULT, est une chaîne de longueur zéro ({}).

### NextChainAreaPtr (PMQACH)-entrée

Un pointeur vers la zone de chaîne d'exit suivante avec une valeur initiale, définie par MQACH\_DEFAULT, de pointeur null (NULL).

Les fonctions de sortie doivent libérer le stockage pour toutes les zones de chaîne de sortie qu'elles acquièrent et manipuler les pointeurs de chaîne pour supprimer leurs zones de chaîne de sortie de la liste.

Une zone de chaîne de sortie peut être construite comme suit:



## Constantes externes

Utilisez cette rubrique comme informations de référence pour les constantes externes disponibles pour l'API.

Les constantes externes suivantes sont disponibles pour les exits API:

### MQXF\_\* (identificateurs de fonction d'exit)

MQXF_INIT	1	X'00000001'
MQXF_TERM	2	X'00000002'
MQXF_CONN	3	X'00000003'
MQXF_CONNX	4	X'00000004'
MQXF_DISC	5	X'00000005'
MQXF_OPEN	6	X'00000006'
MQXF_CLOSE	7	X'00000007'
MQXF_PUT1	8	X'00000008'
MQXF_PUT	9	X'00000009'
MQXF_GET	10	X'0000000A'
MQXF_DATA_CONV_ON_GET	11	X'0000000B'
MQXF_INQ	12	X'0000000C'
MQXF_SET	13	X'0000000D'
MQXF_BEGIN	14	X'0000000E'
MQXF_CMIT	15	X'0000000F'
MQXF_BACK	16	X'00000010'
MQXF_STAT	18	X'00000012'
MQXF_CB	19	X'00000013'
MQXF_CTL	20	X'00000014'
MQXF_CALLBACK	21	X'00000015'
MQXF_SUB	22	X'00000016'
MQXF_SUBRQ	23	X'00000017'
MQXF_XACLOSE	24	X'00000018'
MQXF_XACOMMIT	25	X'00000019'
MQXF_XACOMplete	26	X'0000001A'
MQXF_XAEND	27	X'0000001B'
MQXF_XAFORGET	28	X'0000001C'
MQXF_XAOPEN	29	X'0000001D'
MQXF_XAPREPARE	30	X'0000001E'
MQXF_XARECOVER	31	X'0000001F'

MQXF_XAROLLBACK	32	X'00000020'
MQXF_XASTART	33	X'00000021'
MQXF_AXREG	34	X'00000022'
MQXF_AXUNREG	35	X'00000023'

### MQXR\_\* (raisons de l'exit)

MQXR_BEFORE	1	X'00000001'
MQXR_AFTER	2	X'00000002'
MQXR_CONNECTION	3	X'00000003'

### MQXE\_\* (environnements)

MQXE_OTHER	0	X'00000000'
MQXE_MCA	1	X'00000001'
MQXE_MCA_SVRCONN	2	X'00000002'
MQXE_COMMAND_SERVER	3	X'00000003'
MQXE_MQSC	4	X'00000004'

### MQ\*\_\* (constantes supplémentaires)

MQAXP_VERSION_1	1	
MQAXP_VERSION_2	2	
MQAXC_VERSION_1	1	
MQACH_VERSION_1	1	
MQAXP_CURRENT_VERSION	1	
MQAXC_CURRENT_VERSION	1	
MQACH_CURRENT_VERSION	1	
MQXACT_EXTERNAL	1	
MQXACT_INTERNAL	2	
MQXT_API_EXIT	2	
MQACH_LENGTH_1	68 (32-bit platforms) 72 (64-bit platforms) 80 (128-bit platforms)	
MQACH_CURRENT_LENGTH	68 (32-bit platforms) 72 (64-bit platforms) 80 (128-bit platforms)	

### MQ\*\_\* (constantes null)

MQXPDA_NONE	X'00...00' (48 nulls)
MQXPDA_NONE_ARRAY	'\0','\0',...,'\0','\0'

### MQXCC\_\* (codes achèvement)

MQXCC_FAILED	-8
--------------	----

### MQRC\_\* (codes anomalie)

#### MQRC\_API\_EXIT\_ERROR 2374 X'00000946'

Un appel de fonction d'exit a renvoyé un code de réponse non valide ou a échoué d'une manière ou d'une autre, et le gestionnaire de files d'attente ne peut pas déterminer l'action suivante à effectuer.

Examinez les zones ExitResponse et ExitResponse2 de MQAXP pour déterminer le code de réponse incorrect et modifiez l'exit pour qu'il renvoie un code de réponse valide.

#### MQRC\_API\_EXIT\_INIT\_ERROR 2375 X'00000947'

Le gestionnaire de files d'attente a rencontré une erreur lors de l'initialisation de l'environnement d'exécution pour une fonction d'exit d'API.

#### **MQRC\_API\_EXIT\_TERM\_ERROR 2376 X'00000948'**

Le gestionnaire de files d'attente a rencontré une erreur lors de la fermeture de l'environnement d'exécution pour une fonction d'exit d'API.

#### **MQRC\_EXIT\_REASON\_ERROR 2377 X'00000949'**

La valeur de la zone ExitReason fournie sur un appel d'enregistrement de point d'entrée d'exit (MQXEP) est erronée.

Examinez la valeur de la zone ExitReason pour déterminer et corriger la valeur de la raison de sortie incorrecte.

#### **MQRC\_RESERVED\_VALUE\_ERROR 2378 X'0000094A'**

La valeur de la zone Réserve est erronée.

Examinez la valeur de la zone Réserve pour déterminer et corriger la valeur Réserve.

## **Types de langage C**

Cette rubrique fournit des informations sur les typedefs associés aux exits API disponibles en langage C.

Voici les définitions de type de langage C associées aux exits API:

```
typedef PMQLONG MQPOINTER PPMQLONG;
typedef PMQBYTE MQPOINTER PPMQBYTE;
typedef PMQHOBJS MQPOINTER PPMQHOBJS;
typedef PMQOD MQPOINTER PPMQOD;
typedef PMQMD MQPOINTER PPMQMD;
typedef PMQPMO MQPOINTER PPMQPMO;
typedef PMQGMO MQPOINTER PPMQGMO;
typedef PMQCNO MQPOINTER PPMQCNO;
typedef PMQB0 MQPOINTER PPMQB0;

typedef MQAXP MQPOINTER PMQAXP;
typedef MQACH MQPOINTER PMQACH;
typedef MQAXC MQPOINTER PMQAXC;

typedef MQCHAR MQCHAR16[16];
typedef MQCHAR16 MQPOINTER PMQCHAR16;

typedef MQLONG MQPID;
typedef MQLONG MQTID;
```

## **L'appel d'enregistrement de point d'entrée d'exit (MQXEP)**

Utilisez ces informations pour en savoir plus sur MQXEP, l'appel de langage C MQXEP et le prototype de fonction C MQXEP.

Utilisez l'appel MQXEP pour:

1. Enregistrer les points d'appel d'exit d'API IBM MQ avant et après auxquels appeler les fonctions d'exit
2. Indiquez les points d'entrée de la fonction d'exit
3. Désenregistrer les points d'entrée de la fonction d'exit

Vous codez généralement les appels MQXEP dans la fonction d'exit MQ\_INIT\_EXIT, mais vous pouvez les spécifier dans n'importe quelle fonction d'exit suivante.

Si vous utilisez un appel MQXEP pour enregistrer une fonction d'exit déjà enregistrée, le deuxième appel MQXEP aboutit et remplace la fonction d'exit enregistrée.

Si vous utilisez un appel MQXEP pour enregistrer une fonction d'exit NULL, l'appel MQXEP aboutit et la fonction d'exit est désenregistrée.

Si des appels MQXEP sont utilisés pour enregistrer, désenregistrer et réenregistrer une fonction d'exit particulière pendant la durée de vie d'une demande de connexion, la fonction d'exit précédemment enregistrée est réactivée. Toute mémoire encore allouée et associée à cette instance de fonction d'exit est disponible pour être utilisée par les fonctions de l'exit. (Cette mémoire est généralement libérée lors de l'appel de la fonction d'exit d'arrêt.)

L'interface de MQXEP est la suivante:

```
MQXEP (Hconfig, ExitReason, Function, EntryPoint, &ExitOpts, &CompCode, &Reason)
```

où :

#### **Hconfig (MQHCONFIG)-entrée**

Descripteur de configuration représentant l'exit API qui inclut l'ensemble de fonctions en cours d'initialisation. Cette valeur est générée par le gestionnaire de files d'attente immédiatement avant l'appel de la fonction MQ\_INIT\_EXIT et est transmise dans MQAXP à chaque fonction d'exit d'API.

#### **ExitReason (MQLONG)-entrée**

Motif pour lequel le point d'entrée est enregistré, pour les raisons suivantes:

- Initialisation ou arrêt du niveau de connexion (MQXR\_CONNECTION)
- Avant un appel API IBM MQ (MQXR\_BEFORE)
- Après un appel d'API IBM MQ (MQXR\_AFTER)

#### **Fonction (MQLONG)-entrée**

Identificateur de fonction, dont les valeurs valides sont les constantes MQXF\_ \* (voir «Constantes externes», à la page 1649).

#### **EntryPoint (PMQFUNC)-entrée**

Adresse du point d'entrée de la fonction d'exit à enregistrer. La valeur NULL indique que la fonction d'exit n'a pas été fournie ou qu'un enregistrement précédent de la fonction d'exit est en cours de désenregistrement.

#### **ExitOpts(MQXEPO)**

Les exits API peuvent spécifier des options qui contrôlent la manière dont les exits API sont enregistrés. Si un pointeur null est spécifié pour cette zone, les valeurs par défaut de la structure MQXEPO sont utilisées.

#### **CompCode (MQLONG)-sortie**

Le code achèvement, dont les valeurs valides sont les suivantes:

##### **MQCC\_OK**

Achèvement réussi.

##### **MQCC\_FAILED**

Echec de l'appel.

#### **Cause (MQLONG)-sortie**

Code anomalie qui qualifie le code achèvement.

Si le code achèvement est MQCC\_OK:

##### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC\_FAILED:

##### **ERREUR MQRC\_HCONFIG\_ERROR**

(2280, X'8E8') Le descripteur de configuration fourni n'est pas valide. Utilisez le descripteur de configuration de MQAXP.

##### **MQRC\_EXIT\_MOTIF-ERREUR**

(2377, X' 949') La raison d'appel de la fonction d'exit fournie n'est pas valide ou n'est pas valide pour l'identificateur de fonction d'exit fourni.

Utilisez l'une des raisons d'appel de fonction d'exit valides (valeur MQXR\_\*) ou utilisez une combinaison d'identificateur de fonction et de raison d'exit valide. (Voir [Tableau 836](#), à la page 1653.)

##### **Erreur de fonction MQRC\_FUNCTION\_ERROR**

(2281, X'8E9') L'identificateur de fonction fourni n'est pas valide pour la raison de l'exit API. Le tableau suivant présente les combinaisons valides d'identificateurs de fonction et de ExitReasons.

Tableau 836. Combinaisons valides d'identificateurs de fonction et de ExitReasons	
Fonction	ExitReason
MQXF_INIT MQXF_TERME	MQXR_CONNECTION
MQXF_CONN MQXF_CONNX MQXF_DISQUE MQXF_OPEN MQXF_FERMETURE MQXF_PUT1 MQXF_PUT MQXF_GET MQXF_INQ MQXF_SET MQXF_DÉBUT MQXF_CMIT MQXF_RETOUR MQXF_STAT MQXF_CB MQXF_CTL MQXF_RAPPEL MQXF_SUB MQXF_SUBRQ	MQXR_AVANT MQXR_APRES
MQXF_DATA_CONV_ON_GET	MQXR_AVANT

#### **MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') Une tentative d'enregistrement ou de désenregistrement d'une fonction d'exit a échoué en raison d'un problème de ressource.

#### **MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') Une tentative d'enregistrement ou de désenregistrement d'une fonction d'exit a échoué de manière inattendue.

#### **MQRC\_PROPERTY\_NAME\_ERROR**

(2442, X'098A') Nom ExitProperties non valide.

#### **MQRC\_XEPO\_ERREUR**

(2507, X'09CB') La structure des options d'exit n'est pas valide.

### **Appel du langage C MQXEP**

```
MQXEP (Hconfig, ExitReason, Function, EntryPoint, &ExitOpts, &CompCode, &Reason);
```

Déclaration pour la liste de paramètres:

```
MQHCONFIG      Hconfig;          /* Configuration handle */
MQLONG         ExitReason;       /* Exit reason */
MQLONG         Function;         /* Function identifier */
PMQFUNC       EntryPoint;       /* Function entry point */
MQXEPO         ExitOpts;        /* Options that control the action of MQXEP */
MQLONG         CompCode;        /* Completion code */
MQLONG         Reason;          /* Reason code qualifying completion
                                code */
```

## Prototype de fonction C MQXEP

```
void MQXEP (
MQHCONFIG Hconfig,      /* Configuration handle */
MQLONG ExitReason,     /* Exit reason */
MQLONG Function,       /* Function identifier */
PMQFUNC EntryPoint,    /* Function entry point */
PMQXEPO pExitOpts;     /* Options that control the action of MQXEP */
MQLONG pCompCode,      /* Address of completion code */
MQLONG pReason);       /* Address of reason code qualifying completion
                        code */
```

## Fonctions de sortie

Cette section fournit des informations générales pour vous aider lors de l'utilisation des appels de fonction et décrit comment appeler les fonctions d'exit individuelles.

Utilisez ces informations pour comprendre les règles générales applicables aux routines d'exit d'API, ainsi que pour configurer et nettoyer l'environnement d'exécution de l'exit.

## Règles générales pour les routines d'exit d'API

Les règles générales suivantes s'appliquent lors de l'appel de routines d'exit d'API:

- Dans tous les cas, les fonctions d'exit d'API sont gérées avant la validation des paramètres d'appel d'API et avant les contrôles de sécurité (dans le cas de MQCONN, MQCONNX ou MQOPEN).
- Les valeurs des zones entrées dans et en sortie d'une routine d'exit sont les suivantes:
  - Lors de l'entrée d'une fonction d'exit d'API *avant* IBM MQ , la valeur d'une zone peut être définie par le programme d'application ou par un appel de fonction d'exit précédent.
  - Dans la sortie d'une fonction d'exit d'API *avant* IBM MQ , la valeur d'une zone peut rester inchangée ou être définie sur une autre valeur par la fonction d'exit.
  - En entrée d'une fonction d'exit d'API *après* IBM MQ , la valeur d'une zone peut être la valeur définie par le gestionnaire de files d'attente après le traitement de l'appel d'API IBM MQ , ou peut être définie sur une valeur par un appel de fonction d'exit précédent dans la chaîne de fonctions d'exit.
  - Dans la sortie d'une fonction d'exit d'appel d'API *après* IBM MQ , la valeur d'une zone peut rester inchangée ou être définie sur une autre valeur par la fonction d'exit.
- Les fonctions d'exit doivent communiquer avec le gestionnaire de files d'attente à l'aide des zones ExitResponse et ExitResponse2 .
- Les zones CompCode et Code anomalie communiquent avec l'application. Les fonctions de gestionnaire de files d'attente et d'exit peuvent définir les zones CompCode et Code anomalie.
- L'appel MQXEP renvoie de nouveaux codes anomalie aux fonctions d'exit qui appellent MQXEP. Toutefois, les fonctions de sortie peuvent convertir ces nouveaux codes raison en codes raison existants que les applications existantes et nouvelles peuvent comprendre.
- Chaque prototype de fonction d'exit possède des paramètres similaires à la fonction d'API avec un niveau supplémentaire d'adressage indirect, à l'exception de CompCode et de Reason.
- Les exits API peuvent émettre des appels MQI (à l'exception de MQDISC), mais ces appels MQI n'appellent pas eux-mêmes les exits API.

Notez que, que l'application se trouve sur un serveur ou un client, vous ne pouvez pas prévoir le séquençement des appels d'exit d'API. Un appel BEFORE d'exit API peut ne pas être immédiatement suivi d'un appel AFTER .

L'appel BEFORE peut être suivi d'un autre appel BEFORE . Exemple :

```
AVANT MQCTL
AVANT le rappel
AVANT MQPUT
APRES MQPUT
```

Rappel APRES  
APRÈS MQCTL

ou

AVANT XAOPEN  
AVANT MQCONN  
APRES MQCONN  
APRÈS XAOPEN

Sur le client, il existe un exit qui peut modifier le comportement de l'appel MQCONN ou MQCONNX, appelé exit `PreConnect`. L'exit `PreConnect` peut modifier n'importe quel paramètre de l'appel MQCONN ou MQCONNX, y compris le nom du gestionnaire de files d'attente. Le client appelle d'abord cet exit, puis appelle l'appel MQCONN ou MQCONNX. Notez que seul l'appel MQCONN ou MQCONNX initial appelle l'exit API ; les appels de reconnexion ultérieurs n'ont aucun effet.

## Environnement d'exécution

En général, toutes les erreurs des fonctions d'exit sont communiquées au gestionnaire d'exit à l'aide des zones `ExitResponse` et `ExitResponse2` de MQAXP.

Ces erreurs sont à leur tour converties en valeurs MQCC\_\* et MQRC\_\* et communiquées à l'application dans les zones `CompCode` et `Motif`. Toutefois, toutes les erreurs rencontrées dans la logique du gestionnaire d'exit sont communiquées à l'application sous forme de valeurs MQCC\_\* et MQRC\_\* dans les zones `CompCode` et `Motif`.

Si une fonction MQ\_TERM\_EXIT renvoie une erreur:

- L'appel MQDISC a déjà eu lieu
- Il n'y a pas d'autre possibilité de piloter la *après la fonction d'exit* MQ\_TERM\_EXIT (et donc d'effectuer un nettoyage de l'environnement d'exécution d'exit)
- Le nettoyage de l'environnement d'exécution de sortie n'est pas effectué

L'exit ne peut pas être déchargé car il est peut-être encore en cours d'utilisation. En outre, les autres exits enregistrés plus bas dans la chaîne d'exit pour laquelle l'exit *avant* a abouti seront gérés dans l'ordre inverse.

## Configuration de l'environnement d'exécution de l'exit

Lors du traitement d'un appel MQCONN ou MQCONNX explicite, la logique de traitement de l'exit configure l'environnement d'exécution de l'exit avant d'appeler la fonction d'initialisation de l'exit (MQ\_INIT\_EXIT). La configuration de l'environnement d'exécution d'exit implique le chargement de l'exit, l'acquisition de la mémoire et l'initialisation des structures de paramètres d'exit. Le descripteur de configuration d'exit est également alloué.

Si des erreurs se produisent au cours de cette phase, l'appel MQCONN ou MQCONNX échoue avec `CompCode` MQCC\_FAILED et l'un des codes anomalie suivants:

### **MQRC\_API\_EXIT\_LOAD\_ERROR**

Une tentative de chargement d'un module d'exit API a échoué.

### **MQRC\_API\_EXIT\_NOT\_FOUND**

Une fonction d'exit d'API est introuvable dans le module d'exit d'API.

### **MQRC\_STORAGE\_NOT\_AVAILABLE**

Une tentative d'initialisation de l'environnement d'exécution pour une fonction d'exit d'API a échoué car la mémoire disponible était insuffisante.

### **MQRC\_API\_EXIT\_INIT\_ERROR**

Une erreur s'est produite lors de l'initialisation de l'environnement d'exécution pour une fonction d'exit d'API.

## Nettoyage de l'environnement d'exécution de l'exit

Lors du traitement d'un appel MQDISC explicite ou d'une demande de déconnexion implicite suite à l'arrêt d'une application, la logique de traitement de l'exit peut avoir besoin de nettoyer l'environnement d'exécution de l'exit après avoir appelé la fonction d'arrêt de l'exit (MQ\_TERM\_EXIT), si elle est enregistrée.

Le nettoyage de l'environnement d'exécution d'exit implique la libération de la mémoire pour les structures de paramètres d'exit, éventuellement la suppression des modules précédemment chargés en mémoire.

Si des erreurs se produisent au cours de cette phase, un appel MQDISC explicite échoue avec CompCode MQCC\_FAILED et le code anomalie suivant (les erreurs ne sont pas mises en évidence sur les demandes de déconnexion implicites):

### **MQRC\_API\_EXIT\_TERM\_ERREUR**

Une erreur s'est produite lors de la fermeture de l'environnement d'exécution pour une fonction d'exit d'API. L'exit ne doit pas renvoyer d'échec à partir du MQDISC avant ou après les appels de la fonction d'exit d'API MQ\_TERM\*.

### **Exits d'API sur les clients**

Un client utilise l'exit PreConnect pour modifier le comportement des appels MQCONN et MQCONNX et ne prend pas en charge les propriétés d'exit d'API.

### **Exit PreConnect**

Sur un client, l'exit PreConnect peut être utilisé pour rechercher la définition de canal à partir d'un référentiel central, tel qu'un serveur LDAP.

L'exit PreConnect peut également modifier n'importe quel paramètre, ou tous les paramètres, sur un appel MQCONN ou MQCONNX lui-même, par exemple, le nom du gestionnaire de files d'attente.

Dans le cas des applications client, l'exit PreConnect doit être appelé avant l'exit API car l'exit API MQCONN ou MQCONNX est appelé uniquement une fois que le nom du gestionnaire de files d'attente est connu et que ce nom peut être modifié par l'exit PreConnect .

Notez que seul l'appel MQCONN ou MQCONNX initial appelle l'exit.

### **Propriétés de l'exit API**

Sur un serveur, les exits API peuvent enregistrer une structure MQXEPO lors de l'initialisation. La structure MQXEPO contient la zone ExitProperties qui détaille le groupe de propriétés qui intéresse l'exit. Cela a pour effet de générer un descripteur de propriété de message distinct que l'exit peut manipuler séparément de tout descripteur de propriété de message d'application.

Sur un client, les propriétés d'exit d'API ne sont pas prises en charge. Si une tentative d'enregistrement d'un nom de groupe de propriétés est effectuée sur un client, la fonction échoue avec le code anomalie MQRC\_EXIT\_PROPS\_NOT\_SUPPORTED.

### **Annulation-MQ\_BACK\_EXIT**

MQ\_BACK\_EXIT fournit une fonction d'exit d'annulation permettant d'exécuter *avant* et *après* le traitement d'annulation. Utilisez l'identificateur de fonction MQXF\_BACK avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer les fonctions d'exit d'appel d'annulation *avant* et *après* .

L'interface de cette fonction est:

```
MQ_BACK_EXIT (&ExitParms, &ExitContext, &Hconn, &CompCode, &Reason)
```

où les paramètres sont :

#### **ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.



### ExitContext (MQAXC)-entrée/sortie

Quittez la structure de contexte.

### Hconn (MQHCONN)-entrée

Descripteur de connexion.

### CompCode (MQLONG)-entrée/sortie

Code achèvement, valeurs valides pour lesquelles:

#### MQCC\_OK

Achèvement réussi.

#### MQCC\_WARNING

Achèvement partiel.

#### MQCC\_FAILED

Echec de l'appel

### Motif (MQLONG)-entrée / sortie

Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC\_OK, la seule valeur valide est:

#### MQRC\_NONE

(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC\_FAILED ou MQCC\_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC\_ \* valide.

## Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP    ExitParms;      /* Exit parameter structure */
MQAXC    ExitContext;    /* Exit context structure */
MQHCONN  Hconn;          /* Connection handle */
MQLONG   CompCode;       /* Completion code */
MQLONG   Reason;         /* Reason code qualifying completion code */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_BACK_EXIT (&ExitParms, &ExitContext, &Hconn, &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
void MQENTRY MQ_BACK_EXIT (
PMQAXP    pExitParms,      /* Address of exit parameter structure */
PMQAXC    pExitContext,    /* Address of exit context structure */
PMQHCONN  pHconn,          /* Address of connection handle */
PMQLONG   pCompCode,       /* Address of completion code */
PMQLONG   pReason);        /* Address of reason code qualifying completion
                             code */
```

### Début-MQ\_BEGIN\_EXIT

MQ\_BEGIN\_EXIT fournit une fonction d'exit de début permettant d'exécuter *avant* et *après* le traitement de l'appel MQBEGIN. Utilisez l'identificateur de fonction MQXF\_BEGIN avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer *avant* et *après* les fonctions d'exit d'appel MQBEGIN.

L'interface de cette fonction est:

```
MQ_BEGIN_EXIT (&ExitParms, &ExitContext, &Hconn, &pBeginOptions, &CompCode,
               &Reason)
```

où les paramètres sont :

**ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

**ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

**Hconn (MQHCONN)-entrée**

Descripteur de connexion.

**Options pBegin(PMQBO)-entrée / sortie**

Pointeur vers les options de début.

**CompCode (MQLONG)-entrée/sortie**

Code achèvement, valeurs valides pour lesquelles:

**MQCC\_OK**

Achèvement réussi.

**MQCC\_WARNING**

Achèvement partiel.

**MQCC\_FAILED**

Echec de l'appel

**Motif (MQLONG)-entrée / sortie**

Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC\_OK, la seule valeur valide est:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC\_FAILED ou MQCC\_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC\_\* valide.

**Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP      ExitParms;      /* Exit parameter structure */
MQAXC      ExitContext;    /* Exit context structure */
MQHCONN    Hconn;         /* Connection handle */
PMQBO      pBeginOptions; /* Ptr to begin options */
MQLONG     CompCode;      /* Completion code */
MQLONG     Reason;        /* Reason code qualifying completion code */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_BEGIN_EXIT (&ExitParms, &ExitContext, &Hconn, &pBeginOptions, &CompCode,
               &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
void MQENTRY MQ_BEGIN_EXIT (
PMQAXP      pExitParms,    /* Address of exit parameter structure */
PMQAXC      pExitContext,  /* Address of exit context structure */
PMQHCONN    pHconn,       /* Address of connection handle */
PPMQBO      ppBeginOptions, /* Address of ptr to begin options */
PMQLONG     pCompCode,     /* Address of completion code */
PMQLONG     pReason);     /* Address of reason code qualifying completion
                           code */
```

**Rappel-MQ\_CALLBACK\_EXIT**

MQ\_CALLBACK\_EXIT fournit une fonction d'exit permettant d'exécuter *avant* et *après* le traitement du rappel. Utilisez l'identificateur de fonction MQXF\_CALLBACK avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer les fonctions d'exit d'appel de rappel *avant* et *après*.

L'interface de cette fonction est:

```
MQ_CALLBACK_EXIT (&ExitParms, &ExitContext, &Hconn, &pMsgDesc, &pGetMsgOpts,  
                  &pBuffer, &pMQCBCContext)
```

où les paramètres sont :

**ExitParms (MQAXP)-entrée/sortie**

Structure de paramètre d'exit

**ExitContext (MQAXC)-entrée/sortie**

Structure de contexte de sortie

**Hconn (MQHCONN)-entrée/sortie**

Descripteur de connexion

**pMsgDescription**

Descripteur de message

**pGetMsgOpts**

Options qui contrôlent l'action de MQGET

**pBuffer**

Zone devant contenir les données de message

**pMQCBCContext**

Données contextuelles pour le rappel

## Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP      ExitParms;      /* Exit parameter structure */  
MQAXC      ExitContext;    /* Exit context structure */  
MQHCONN    Hconn;         /* Connection handle */  
PMQMD      pMsgDesc;      /* Message descriptor */  
PMQGMO     pGetMsgOpts;   /* Options that define the operation of the consumer */  
PMQVOID    pBuffer;       /* Area to contain the message data */  
PMQCBC     pContext;      /* Context data for the callback */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_SUBRQ_EXIT (&ExitParms, &ExitContext, &Hconn, &pMsgDesc, &pGetMsgOpts, &pBuffer,  
               &pContext);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
void MQENTRY MQ_CALLBACK_EXIT (  
PMQAXP      pExitParms;    /* Exit parameter structure */  
PMQAXC      pExitContext;  /* Exit context structure */  
PMQHCONN    pHconn;       /* Connection handle */  
PPMQMD      ppMsgDesc;    /* Message descriptor */  
PPMQGMO     ppGetMsgOpts; /* Options that define the operation of the consumer */  
PPMQVOID    ppBuffer;     /* Area to contain the message data */  
PPMQCBC     ppContext;    /* Context data for the callback */
```

## Remarques sur l'utilisation

1. L'exit de rappel est appelé avant l'appel du consommateur et après la fin de la fonction de consommateur du consommateur. Bien que les structures MQMD et MQGMO puissent être modifiées, la modification des valeurs dans l'exit avant ne reconduit pas l'extraction d'un message de la file d'attente car le message a déjà été supprimé de la file d'attente pour être distribué à la fonction de destinataire

## Gestion des fonctions de rappel-MQ\_CB\_EXIT

MQ\_CB\_EXIT fournit une fonction d'exit permettant d'exécuter *avant* et *après* l'appel MQCB. Utilisez l'identificateur de fonction MQXF\_CB avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer *avant* et *après* les fonctions d'exit d'appel MQCB.

L'interface de cette fonction est:

```
MQ_CB_EXIT (&ExitParms, &ExitContext, &Hconn, &Operation, &pCallbackDesc,  
           &Hobj, &pMsgDesc, &pGetMsgOpts, &CompCode, &Reason)
```

où les paramètres sont :

### ExitParms (MQAXP)-entrée/sortie

Structure de paramètre d'exit

### ExitContext (MQAXC)-entrée/sortie

Structure de contexte de sortie

### Hconn (MQHCONN)-entrée/sortie

Descripteur de connexion

### Opération (MQLONG)-entrée / sortie

Valeur de l'opération

### pCallbackDesc (PMQCBD)-entrée / sortie

Descripteur de rappel

### Hobj (MQHOBJ)-entrée/sortie

Descripteur d'objet

### pMsgDescription (PMQMD)-entrée/sortie

Descripteur de message

### pGetMsgOpts (PMQGMO)-entrée/sortie

Options qui contrôlent l'action de MQCB

### CompCode (MQLONG)-entrée/sortie

Code de fin d'exécution

### Motif (MQLONG)-entrée / sortie

Code anomalie qualifiant CompCode

## Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP      ExitParms;      /* Exit parameter structure */  
MQAXC      ExitContext;    /* Exit context structure */  
MQHCONN    Hconn;         /* Connection handle */  
MQLONG     Operation;     /* Operation value. */  
MQCBD      pMsgDesc;      /* Callback descriptor. */  
MQHOBJ     Hobj;         /* Object handle. */  
PMQMD      pMsgDesc;      /* Message descriptor */  
PMQGMO     pGetMsgOpts;   /* Options that define the operation of the consumer */  
PMQLONG    CompCode;      /* Completion code. */  
PMQLONG    Reason;        /* Reason code qualifying CompCode. */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_CB_EXIT (&ExitParms, &ExitContext, &Hconn, &Operation, &Hobj, &pMsgDesc,  
           &pGetMsgOpts, &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
void MQENTRY MQ_CB_EXIT (  
PMQAXP      pExitParms;    /* Exit parameter structure */  
PMQAXC      pExitContext;  /* Exit context structure */  
PMQHCONN    pHconn;       /* Connection handle */
```

```

PMQLONG    pOperation;      /* Callback operation */
PMQHOBJS   pHobj;          /* Object handle */
PPMQMD     ppMsgDesc;      /* Message descriptor */
PPMQGMO    ppGetMsgOpts;   /* Options that control the action of MQCB */
PMQLONG    pCompCode;      /* Completion code */
PMQLONG    pReason;        /* Reason code qualifying CompCode */

```

### **Fermer-MQ\_CLOSE\_EXIT**

MQ\_CLOSE\_EXIT fournit une fonction d'exit de fermeture permettant d'exécuter *avant* et *après* le traitement des appels MQCLOSE. Utilisez l'identificateur de fonction MQXF\_CLOSE avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer les fonctions d'exit d'appel MQCLOSE *avant* et *après*.

L'interface de cette fonction est:

```

MQ_CLOSE_EXIT (&ExitParms, &ExitContext, &Hconn, &pHobj,
               &Options, &CompCode, &Reason)

```

où les paramètres sont :

#### **ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

#### **ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

#### **Hconn (MQHCONN)-entrée**

Descripteur de connexion.

#### **pHobj (PMQHOBJS)-entrée**

Pointeur vers le descripteur d'objet.

#### **Options (MQLONG)-entrée / sortie**

Options de fermeture.

#### **CompCode (MQLONG)-entrée/sortie**

Code achèvement, valeurs valides pour lesquelles:

##### **MQCC\_OK**

Achèvement réussi.

##### **MQCC\_FAILED**

Echec de l'appel

#### **Motif (MQLONG)-entrée / sortie**

Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC\_OK, la seule valeur valide est:

##### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC\_FAILED, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC\_\* valide.

### **Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```

MQAXP      ExitParms;      /* Exit parameter structure */
MQAXC      ExitContext;    /* Exit context structure */
MQHCONN    Hconn;          /* Connection handle */
PMQHOBJS   pHobj;         /* Ptr to object handle */
MQLONG     Options;        /* Close options */
MQLONG     CompCode;       /* Completion code */
MQLONG     Reason;        /* Reason code */

```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_CLOSE_EXIT (&ExitParms, &ExitContext,&Hconn, &pHobj, &Options,
                &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
void MQENTRY MQ_CLOSE_EXIT (
PMQAXP      pExitParms,      /* Address of exit parameter structure */
PMQAXC      pExitContext,    /* Address of exit context structure */
PMQHCONN    pHconn,         /* Address of connection handle */
PPMHOBJS    pHobj,         /* Address of ptr to object handle */
PMQLONG     pOptions,        /* Address of close options */
PMQLONG     pCompCode,       /* Address of completion code */
PMQLONG     pReason);       /* Address of reason code qualifying
                             completion code */
```

### **Validation-MQ\_CMIT\_EXIT**

MQ\_CMIT\_EXIT fournit une fonction d'exit de validation permettant d'exécuter *avant* et *après* le traitement de validation. Utilisez l'identificateur de fonction MQXF\_CMIT avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer les fonctions d'exit d'appel de validation *avant* et *après*.

Si une opération de validation échoue et que la transaction est annulée, l'appel MQCMIT échoue avec MQCC\_WARNING et MQRC\_BACKED\_OUT. Ces codes retour et anomalie sont transmis à toutes les fonctions d'exit *après* MQCMIT pour indiquer aux fonctions d'exit que l'unité de travail a été annulée.

L'interface de cette fonction est:

```
MQ_CMIT_EXIT (&ExitParms, &ExitContext, &Hconn, &CompCode, &Reason)
```

où les paramètres sont :

#### **ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

#### **ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

#### **Hconn (MQHCONN)-entrée**

Descripteur de connexion.

#### **CompCode (MQLONG)-entrée/sortie**

Code achèvement, valeurs valides pour lesquelles:

##### **MQCC\_OK**

Achèvement réussi.

##### **MQCC\_WARNING**

Achèvement partiel.

##### **MQCC\_FAILED**

Echec de l'appel

#### **Motif (MQLONG)-entrée / sortie**

Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC\_OK, la seule valeur valide est:

##### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC\_FAILED ou MQCC\_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC\_\* valide.

### **Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```

MQAXP   ExitParms;      /* Exit parameter structure */
MQAXC   ExitContext;   /* Exit context structure */
MQHCONN Hconn;         /* Connection handle */
MQLONG  CompCode;     /* Completion code */
MQLONG  Reason;       /* Reason code qualifying completion code */

```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_CMITY_EXIT (&ExitParms, &ExitContext,&Hconn, &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```

void MQENTRY MQ_CMITY_EXIT (
PMQAXP   pExitParms,      /* Address of exit parameter structure */
PMQAXC   pExitContext,   /* Address of exit context structure */
PMQHCONN pHconn,        /* Address of connection handle */
PMQLONG  pCompCode,     /* Address of completion code */
PMQLONG  pReason);     /* Address of reason code qualifying completion
                        code */

```

## Remarques sur l'utilisation

1. L'interface de fonction MQ\_GET\_EXIT décrite ici est utilisée à la fois pour la fonction d'exit MQXF\_GET et la fonction d'exit «MQXF\_DATA\_CONV\_ON\_GET», à la page 1669 .

Des points d'entrée distincts sont définis pour ces deux fonctions d'exit. Par conséquent, pour intercepter *les deux*, l'appel MQXEP doit être utilisé deux fois ; pour cet appel, utilisez l'identificateur de fonction MQXF\_GET.

L'interface MQ\_GET\_EXIT étant la même pour MQXF\_GET et MQXF\_DATA\_CONV\_ON\_GET, une seule fonction d'exit peut être utilisée pour les deux ; la zone *Function* de la structure MQAXP indique quelle fonction d'exit a été appelée. L'appel MQXEP peut également être utilisé pour enregistrer des fonctions d'exit différentes pour les deux cas.

### **Connexion et extension de connexion-MQ\_CONNX\_EXIT**

MQ\_CONNX\_EXIT fournit une fonction d'exit de connexion pour exécuter *avant* et *après* le traitement MQCONN, et une fonction d'exit d'extension de connexion pour exécuter *avant* et *après* le traitement MQCONNX.

La même interface, décrite ici, est appelée pour les deux fonctions d'exit d'appel MQCONN et MQCONNX.

Lorsque l'agent MCA répond à une connexion client entrante, il peut se connecter et effectuer un certain nombre d'appels API IBM MQ avant que l'état du client ne soit complètement connu. Ces appels API appellent les fonctions d'exit API avec MQAXC en fonction du programme MCA lui-même (par exemple, dans les zones UserId et ConnectionName de MQAXC).

Lorsque l'agent MCA répond aux appels API client entrants suivants, la structure MQAXC est basée sur le client entrant, en définissant les zones UserId et ConnectionName de manière appropriée.

Le nom du gestionnaire de files d'attente défini par l'application sur un appel MQCONN ou MQCONNX est transmis à l'appel de connexion sous-jacent. Toute tentative de modification du nom du gestionnaire de files d'attente par un *avant* MQ\_CONNX\_EXIT n'a aucun effet.

Utilisez les identificateurs de fonction MQXF\_CONN et MQXF\_CONNX avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer *avant* et *après* les fonctions d'exit d'appel MQCONN et MQCONNX.

Un exit MQ\_CONNX\_EXIT appelé pour la raison suivante: MQXR\_BEFORE *ne doit pas* émettre d'appels d'API IBM MQ, car l'environnement correct n'a pas été configuré pour le moment.

Un MQ\_CONNX\_EXIT ne peut pas appeler MQDISC à partir d'un appel d'exit API pour la connexion pour laquelle il est appelé. Cette restriction s'applique aux exits API client et serveur.

L'interface de MQCONN et de MQCONNX est identique:

```
MQ_CONNX_EXIT (&ExitParms, &ExitContext, &pQMgrName, &pConnectOpts,  
&pHconn, &CompCode, &Reason);
```

où les paramètres sont :

**ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

**ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

**pQMgrNom (PMQCHAR)-entrée**

Pointeur vers le nom du gestionnaire de files d'attente fourni dans l'appel MQCONN. L'exit ne doit pas modifier ce nom dans l'appel MQCONN ou MQCONN.

**pConnectOpts (PMQCNO)-entrée/sortie**

Pointeur vers les options qui contrôlent l'action de l'appel MQCONN.

Pour plus d'informations, voir «MQCNO-Options de connexion», à la page 325.

Pour la fonction d'exit MQXF\_CONN, pConnectOpts pointe vers la structure d'options de connexion par défaut (MQCNO\_DEFAULT).

**pHconn (PMQHCONN)-entrée**

Pointeur vers le descripteur de connexion.

**CompCode (MQLONG)-entrée/sortie**

Code achèvement, valeurs valides pour lesquelles:

**MQCC\_OK**

Achèvement réussi.

**MQCC\_WARNING**

Avertissement (achèvement partiel)

**MQCC\_FAILED**

Echec de l'appel

**Motif (MQLONG)-entrée / sortie**

Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC\_OK, la seule valeur valide est:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC\_FAILED ou MQCC\_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC\_\* valide.

## Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP      ExitParms;      /* Exit parameter structure */  
MQAXC      ExitContext;    /* Exit context structure */  
PMQCHAR    pQMgrName;     /* Ptr to Queue manager name */  
PMQCNO     pConnectOpts;  /* Ptr to Connection options */  
PMQHCONN   pHconn;       /* Ptr to Connection handle */  
MQLONG     CompCode;     /* Completion code */  
MQLONG     Reason;       /* Reason code */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_CONNX_EXIT (&ExitParms, &ExitContext, &pQMgrName, &pConnectOps,  
                &pHconn, &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:



```

void MQENTRY MQ_CONNX_EXIT (
PMQAXP      pExitParms,      /* Address of exit parameter structure */
PMQAXC      pExitContext,    /* Address of exit context structure */
PPMQCHAR    ppQMgrName,     /* Address of ptr to queue manager name */
PPMQCNO     ppConnectOpts,  /* Address of ptr to connection options */
PPMQHCONN   ppHconn,        /* Address of ptr to connection handle */
PMQLONG     pCompCode,      /* Address of completion code */
PMQLONG     pReason);       /* Address of reason code qualifying
                             completion code */

```

## Remarques sur l'utilisation

1. L'interface de fonction MQ\_CONNX\_EXIT décrite ici est utilisée à la fois pour l'appel MQCONN et pour l'appel MQCONNX. Toutefois, des points d'entrée distincts sont définis pour ces deux appels. Pour intercepter *les deux* appels, l'appel MQXEP doit être utilisé au moins deux fois-une fois avec l'identificateur de fonction MQXF\_CONN, et de nouveau avec MQXF\_CONNX.

L'interface MQ\_CONNX\_EXIT étant la même pour MQCONN et MQCONNX, une seule fonction d'exit peut être utilisée pour les deux appels ; la zone *Function* de la structure MQAXP indique l'appel en cours. L'appel MQXEP peut également être utilisé pour enregistrer des fonctions d'exit différentes pour les deux appels.

2. Lorsqu'un agent MCA répond à une connexion client entrante, il peut émettre un certain nombre d'appels MQ avant que l'état du client ne soit complètement connu. Ces appels MQ entraînent l'appel des fonctions d'exit API avec la structure MQAXC contenant des données relatives à l'agent MCA et non au client (par exemple, l'identificateur utilisateur et le nom de connexion). Toutefois, une fois que l'état du client est entièrement connu, les appels MQ ultérieurs entraînent l'appel des fonctions d'exit API avec les données client appropriées dans la structure MQAXC.
3. Toutes les fonctions d'exit MQXR\_BEFORE sont appelées avant toute validation de paramètre effectuée par le gestionnaire de files d'attente. Il se peut donc que les paramètres ne soient pas valides (y compris les pointeurs non valides pour les adresses des paramètres).  
La fonction MQ\_CONNX\_EXIT est appelée avant toute vérification d'autorisation effectuée par le gestionnaire de files d'attente.
4. La fonction d'exit ne doit pas modifier le nom du gestionnaire de files d'attente spécifié dans l'appel MQCONN ou MQCONNX. Si le nom est modifié par la fonction d'exit, les résultats ne sont pas définis.
5. Une fonction d'exit MQXR\_BEFORE pour MQ\_CONNX\_EXIT ne peut pas émettre d'appels MQ autres que MQXEP.

## Rappel de contrôle-MQ\_CTL\_EXIT

MQ\_CTL\_EXIT fournit une fonction d'exit de demande d'abonnement pour exécuter *avant* et *après le traitement du rappel de contrôle*. Utilisez l'identificateur de fonction MQXF\_CTL avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer *avant* et *après* les fonctions d'exit d'appel de rappel de contrôle.

L'interface de cette fonction est:

```
MQ_CTL_EXIT (&Hconn, &Operation, &ControlOpts, &CompCode, &Reason)
```

où les paramètres sont :

### Hconn (MQHCONN)-entrée/sortie

Descripteur de connexion.

### Entrée / sortie d'opération (MQLONG)

Opération en cours de traitement sur le rappel défini pour le descripteur d'objet spécifié

### Entrée / sortie ControlOpts (MQCTLO)

Options qui contrôlent l'action de MQCTL

### CompCode (MQLONG)-entrée/sortie

Code achèvement, valeurs valides pour lesquelles:

**MQCC\_OK**

Achèvement réussi.

**MQCC\_WARNING**

Achèvement partiel.

**MQCC\_FAILED**

Echec de l'appel

**Motif (MQLONG)-entrée / sortie**

Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC\_OK, la seule valeur valide est:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC\_FAILED ou MQCC\_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC\_\* valide.

**Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQHCONN  Hconn;          /* Connection handle */
MQLONG   Operation;     /* Operation being processed */
MQCTLO   ControlOpts;   /* Options that control the action of MQCTL */
MQLONG   CompCode;      /* Completion code */
MQLONG   Reason;        /* Reason code qualifying completion code */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_CTL_EXIT (&Hconn, &Operation, &ControlOpts, &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
void MQENTRY MQ_CTL_EXIT (
PMQHCONN pHconn;          /* Address of connection handle */
PMQLONG  pOperation;     /* Address of operation being processed */
PMQCTLO  pControlOpts;   /* Address of options that control the action of MQCTL */
PMQLONG  pCompCode;      /* Address of completion code */
PMQLONG  pReason;        /* Address of reason code qualifying completion code */
```

**Déconnexion-MQ\_DISC\_EXIT**

MQ\_DISC\_EXIT fournit une fonction d'exit de déconnexion permettant d'exécuter *avant* et *après* le traitement de l'exit MQDISC. Utilisez l'identificateur de fonction MQXF\_DISC avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer *avant* et *après* les fonctions d'exit d'appel MQDISC.

L'interface de cette fonction est

```
MQ_DISC_EXIT (&ExitParms, &ExitContext, &pHconn,
&CompCode, &Reason);
```

où les paramètres sont :

**ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

**ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

**pHconn (PMQHCONN)-entrée**

Pointeur vers le descripteur de connexion.

Pour l'appel MQDISC précédent, la valeur de cette zone est l'une des suivantes:

- Descripteur de connexion renvoyé lors de l'appel MQCONN ou MQCONNX
- Zéro, pour les environnements dans lesquels un adaptateur spécifique à l'environnement s'est connecté au gestionnaire de files d'attente
- Valeur définie par un appel de fonction d'exit précédent

Pour l'appel MQDISC après, la valeur de cette zone est zéro ou une valeur définie par un appel de fonction d'exit précédent.

### CompCode (MQLONG)-entrée/sortie

Code achèvement, valeurs valides pour lesquelles:

#### MQCC\_OK

Achèvement réussi.

#### MQCC\_WARNING

Achèvement partiel

#### MQCC\_FAILED

Echec de l'appel

### Motif (MQLONG)-entrée / sortie

Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC\_OK, la seule valeur valide est:

#### MQRC\_NONE

(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC\_FAILED ou MQCC\_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC\_ \* valide.

## Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP      ExitParms;      /* Exit parameter structure */
MQAXC      ExitContext;    /* Exit context structure */
PMQHCONN   pHconn;        /* Ptr to Connection handle */
MQLONG     CompCode;       /* Completion code */
MQLONG     Reason;        /* Reason code */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_DISC_EXIT (&ExitParms, &ExitContext, &pHconn,
              &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
void MQENTRY MQ_DISC_EXIT (
PMQAXP      pExitParms,    /* Address of exit parameter structure */
PMQAXC      pExitContext,  /* Address of exit context structure */
PPMHCONN    pHconn,       /* Address of ptr to connection handle */
PMQLONG     pCompCode,     /* Address of completion code */
PMQLONG     pReason);     /* Address of reason code qualifying
                             completion code */
```

### Get-MQ\_GET\_EXIT

MQ\_GET\_EXIT fournit une fonction d'exit d'extraction permettant d'exécuter *avant* et *après* le traitement de l'appel MQGET.

Il existe deux identificateurs de fonction:

1. Utilisez MQXF\_GET avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer les fonctions d'exit d'appel MQGET *avant* et *après*.

2. Pour plus d'informations sur l'utilisation de l'identificateur de fonction MQXF\_DATA\_CONV\_ON\_GET, voir «MQXF\_DATA\_CONV\_ON\_GET», à la page 1669 .

L'interface de cette fonction est:

```
MQ_GET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,  
            &pGetMsgOpts, &BufferLength, &pBuffer, &pDataLength,  
            &CompCode, &Reason)
```

où les paramètres sont :

**ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

**ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

**Hconn (MQHCONN)-entrée**

Descripteur de connexion.

**Hobj (MQHOBJ)-entrée/sortie**

Descripteur d'objet.

**pMsgDescription (PMQMD)-entrée/sortie**

Pointeur vers le descripteur de message.

**pGetMsgOpts (PMQGMO)-entrée/sortie**

Pointeur pour obtenir les options de message.

**BufferLength (MQLONG)-entrée/sortie**

Longueur de la mémoire tampon de messages.

**pBuffer (PMQBYTE)-entrée/sortie**

Pointeur vers la mémoire tampon de messages.

**pDataLength (PMQLONG)-entrée/sortie**

Pointeur vers la zone de longueur de données.

**CompCode (MQLONG)-entrée/sortie**

Code achèvement, valeurs valides pour lesquelles:

**MQCC\_OK**

Achèvement réussi.

**MQCC\_WARNING**

Achèvement partiel.

**MQCC\_FAILED**

Echec de l'appel

**Motif (MQLONG)-entrée / sortie**

Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC\_OK, la seule valeur valide est:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC\_FAILED ou MQCC\_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC\_\* valide.

## Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP      ExitParms;      /* Exit parameter structure */  
MQAXC      ExitContext;    /* Exit context structure */  
MQHCONN    Hconn;         /* Connection handle */  
MQHOBJ     Hobj;          /* Object handle */  
PMQMD      pMsgDesc;      /* Ptr to message descriptor */
```

```

PMQPMO      pGetMsgOpts;    /* Ptr to get message options */
MQLONG      BufferLength; /* Message buffer length */
PMQBYTE     pBuffer;    /* Ptr to message buffer */
PMQLONG     pDataLength; /* Ptr to data length field */
MQLONG      CompCode;   /* Completion code */
MQLONG      Reason;     /* Reason code */

```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```

MQ_GET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
             &pGetMsgOpts, &BufferLength, &pBuffer, &pDataLength,
             &CompCode, &Reason)

```

Votre exit doit correspondre au prototype de fonction C suivant:

```

void MQENTRY MQ_GET_EXIT (
PMQAXP      pExitParms,    /* Address of exit parameter structure */
PMQAXC      pExitContext, /* Address of exit context structure */
PMQHCONN    pHConn,       /* Address of connection handle */
PMQHOBJ     pHObj,        /* Address of object handle */
PPMQMD      ppMsgDesc,    /* Address of ptr to message descriptor */
PPMQGMO     ppGetMsgOpts, /* Address of ptr to get message options */
MQLONG      pBufferLength, /* Address of message buffer length */
PPMQBYTE    ppBuffer,     /* Address of ptr to message buffer */
PPMQLONG    ppDataLength, /* Address of ptr to data length field */
MQLONG      pCompCode,    /* Address of completion code */
MQLONG      pReason);     /* Address of reason code qualifying
                             completion code */

```

## Remarques sur l'utilisation

1. L'interface de fonction MQ\_GET\_EXIT décrite ici est utilisée à la fois pour la fonction d'exit MQXF\_GET et la fonction d'exit «MQXF\_DATA\_CONV\_ON\_GET», à la page 1669.

Des points d'entrée distincts sont définis pour ces deux fonctions d'exit. Par conséquent, pour intercepter *les deux*, l'appel MQXEP doit être utilisé deux fois ; pour cet appel, utilisez l'identificateur de fonction MQXF\_GET.

L'interface MQ\_GET\_EXIT étant la même pour MQXF\_GET et MQXF\_DATA\_CONV\_ON\_GET, une seule fonction d'exit peut être utilisée pour les deux ; la zone *Function* de la structure MQAXP indique quelle fonction d'exit a été appelée. L'appel MQXEP peut également être utilisé pour enregistrer des fonctions d'exit différentes pour les deux cas.

### **MQXF\_DATA\_CONV\_ON\_GET**

L'identificateur de fonction MQXF\_DATA\_CONV\_ON\_GET est utilisé avec MQ\_GET\_EXIT.

Voir MQ\_GET\_EXIT pour plus d'informations sur l'interface de cet appel et un exemple de déclaration en langage C.

## Remarques sur l'utilisation

S'il est enregistré, ce point d'entrée est appelé lorsque des messages arrivent à l'application mais avant qu'une conversion de données ne soit effectuée. Cela peut être utile si l'exit API doit effectuer un traitement, tel que le déchiffrement ou la décompression, avant que le message ne soit transmis à la conversion de données. L'exit peut, si nécessaire, provoquer le contournement de la conversion de données en renvoyant la fonction MQXCC\_SUPPRESS\_FUNCTION; pour plus d'informations, voir la structure MQAXP.

L'enregistrement de ce point d'entrée sur un client a pour effet d'entraîner l'exécution locale de la conversion de données sur la machine client. Pour un fonctionnement correct, il peut donc être nécessaire d'installer les exits de conversion d'application sur le client. Notez que MQXF\_DATA\_CONV\_ON\_GET est également utilisé pour la consommation asynchrone.

Lors de l'utilisation de l'appel MQ\_GET\_EXIT, utilisez MQXF\_DATA\_CONV\_ON\_GET, avec la raison d'exit MQXR\_BEFORE, pour enregistrer une fonction d'exit de conversion de données *before* MQGET.

Il n'existe pas de fonction d'exit MQXR\_AFTER pour MQXF\_DATA\_CONV\_ON\_GET ; la fonction d'exit MQXR\_AFTER pour MQXF\_GET fournit la fonction requise pour le traitement des exits après la conversion des données.

Des points d'entrée distincts sont définis pour l' appel MQ\_GET\_EXIT. Par conséquent, pour intercepter les deux fonctions d'exit, l'appel MQXEP doit être utilisé deux fois ; pour cet appel, utilisez l'identificateur de fonction MQXF\_DATA\_CONV\_ON\_GET.

L'interface MQ\_GET\_EXIT étant la même pour MQXF\_GET et MQXF\_DATA\_CONV\_ON\_GET, une seule fonction d'exit peut être utilisée pour les deux ; la zone *Function* de la structure MQAXP indique quelle fonction d'exit a été appelée. L'appel MQXEP peut également être utilisé pour enregistrer des fonctions d'exit différentes pour les deux cas.

### **Initialisation-MQ\_INIT\_EXIT**

MQ\_INIT\_EXIT fournit une initialisation de niveau de connexion, indiquée en définissant ExitReason dans MQAXP sur MQXR\_CONNECTION.

Lors de l'initialisation, notez ce qui suit:

- La fonction MQ\_INIT\_EXIT appelle MQXEP pour enregistrer les instructions d'API IBM MQ et les points ENTRY et EXIT qui l'intéressent.
- Les exits n'ont pas besoin d'intercepter toutes les instructions d'API IBM MQ . Les fonctions d'exit sont appelées uniquement si un intérêt a été enregistré.
- La mémoire qui doit être utilisée par l'exit peut être acquise lors de son initialisation.
- Si un appel à cette fonction échoue, l'appel MQCONN ou MQCONNX qui l'a appelé échoue également avec un CompCode et une raison qui dépendent de la valeur de la zone ExitResponse dans MQAXP.
- Un exit MQ\_INIT\_EXIT ne doit pas émettre d'appels d'API IBM MQ , car l'environnement approprié n'a pas été configuré pour le moment.
- Si un exit MQ\_INIT\_EXIT échoue avec MQXCC\_FAILED, le gestionnaire de files d'attente renvoie l'appel MQCONN ou MQCONNX qui l'a appelé avec MQCC\_FAILED et MQRC\_API\_EXIT\_ERROR.
- Si le gestionnaire de files d'attente rencontre une erreur lors de l'initialisation de l'environnement d'exécution de la fonction d'exit d'API avant d'appeler le premier exit MQ\_INIT\_EXIT, le gestionnaire de files d'attente renvoie l'appel MQCONN ou MQCONNX qui a appelé MQ\_INIT\_EXIT avec MQCC\_FAILED et MQRC\_API\_EXIT\_INIT\_ERROR.

L'interface de MQ\_INIT\_EXIT est la suivante:

```
MQ_INIT_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

où les paramètres sont :

#### **ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

#### **ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

#### **CompCode (MQLONG)-entrée/sortie**

Pointeur vers le code achèvement, dont les valeurs admises sont:

##### **MQCC\_OK**

Achèvement réussi.

##### **MQCC\_WARNING**

Achèvement partiel.

##### **MQCC\_FAILED**

Echec de l'appel

#### **Motif (MQLONG)-entrée / sortie**

Pointeur vers le code raison qualifiant le code achèvement.

Si le code achèvement est MQCC\_OK, la seule valeur valide est:

## **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC\_FAILED ou MQCC\_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC\_\* valide.

Le CompCode et le motif renvoyés à l'application dépendent de la valeur de la zone ExitResponse dans MQAXP.

## **Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP      ExitParms;      /* Exit parameter structure */
MQAXC      ExitContext;    /* Exit context structure */
MQQLONG    CompCode;      /* Completion code */
MQQLONG    Reason;        /* Reason code */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_INIT_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
void MQENTRY MQ_INIT_EXIT (
PMQAXP      pExitParms,    /* Address of exit parameter structure */
PMQAXC      pExitContext,  /* Address of exit context structure */
PMQLONG     pCompCode,     /* Address of completion code */
PMQLONG     pReason);      /* Address of reason code qualifying
                             completion code */
```

## **Remarques sur l'utilisation**

1. La fonction MQ\_INIT\_EXIT peut émettre l'appel MQXEP pour enregistrer les adresses des fonctions d'exit pour les appels MQ à intercepter. Il n'est pas nécessaire d'intercepter tous les appels MQ, ni d'intercepter les deux appels MQXR\_BEFORE et MQXR\_AFTER. Par exemple, une suite d'exit peut choisir d'intercepter uniquement l'appel MQXR\_BEFORE de MQPUT.
2. Le stockage qui doit être utilisé par les fonctions d'exit dans la suite d'exit peut être acquis par la fonction MQ\_INIT\_EXIT. Les fonctions d'exit peuvent également acquérir du stockage lorsqu'elles sont appelées, selon les besoins. Toutefois, toute la mémoire doit être libérée avant l'arrêt de la suite d'exit; la fonction MQ\_TERM\_EXIT peut libérer la mémoire ou une fonction d'exit appelée précédemment.
3. Si MQ\_INIT\_EXIT renvoie MQXCC\_FAILED dans la zone ExitResponse de MQAXP ou échoue d'une autre manière, l'appel MQCONN ou MQCONNX qui a provoqué l'appel de MQ\_INIT\_EXIT échoue également, avec les paramètres **CompCode** et **Reason** définis sur les valeurs appropriées.
4. Une fonction MQ\_INIT\_EXIT ne peut pas émettre d'appels MQ autres que MQXEP.

## **Interroger-MQ\_INQ\_EXIT**

MQ\_INQ\_EXIT fournit une fonction d'exit d'interrogation permettant d'exécuter *avant* et *après* le traitement de l'appel MQINQ. Utilisez l'identificateur de fonction MQXF\_INQ avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer les fonctions d'exit d'appel MQINQ *avant* et *après*.

L'interface de cette fonction est:

```
MQ_INQ_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
             &pCharAttrs, &CompCode, &Reason)
```

où les paramètres sont :

**ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

**ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

**Hconn (MQHCONN)-entrée**

Descripteur de connexion.

**Hobj (MQHOBJ)-entrée**

Descripteur d'objet.

**SelectorCount (MQLONG)-entrée**

Nombre de sélecteurs

**pSelectors (PMQLONG)-entrée/sortie**

Pointeur vers un tableau de valeurs de sélecteur.

**IntAttrCount (MQLONG)-entrée**

Nombre d'attributs de type entier.

**Attributs pInt(PMQLONG)-entrée/sortie**

Pointeur vers un tableau de valeurs d'attribut entières.

**CharAttrLongueur (MQLONG)-entrée/sortie**

Longueur du tableau d'attributs de caractères.

**pCharAttrs (PMQCHAR)-entrée / sortie**

Pointeur vers un tableau d'attributs de caractères.

**CompCode (MQLONG)-entrée/sortie**

Code achèvement, valeurs valides pour lesquelles:

**MQCC\_OK**

Achèvement réussi.

**MQCC\_WARNING**

Achèvement partiel.

**MQCC\_FAILED**

Echec de l'appel

**Motif (MQLONG)-entrée / sortie**

Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC\_OK, la seule valeur valide est:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC\_FAILED ou MQCC\_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC\_ \* valide.

**Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP      ExitParms;          /* Exit parameter structure */
MQAXC      ExitContext;       /* Exit context structure */
MQHCONN    Hconn;            /* Connection handle */
MQHOBJ     Hobj;              /* Object handle */
MQLONG     SelectorCount;     /* Count of selectors */
PMQLONG    pSelectors;        /* Ptr to array of attribute selectors */
MQLONG     IntAttrCount;      /* Count of integer attributes */
PMQLONG    pIntAttrs;         /* Ptr to array of integer attributes */
MQLONG     CharAttrLength;    /* Length of char attributes array */
PMQCHAR    pCharAttrs;       /* Ptr to character attributes */
MQLONG     CompCode;          /* Completion code */
MQLONG     Reason;           /* Reason code qualifying completion code */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:



```
MQ_INQ_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
             &pCharAttrs, &CompCode, &Reason)
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
void MQENTRY MQ_INQ_EXIT (
PMQAXP   pExitParms,      /* Address of exit parameter structure */
PMQAXC   pExitContext,    /* Address of exit context structure */
PMQHCONN pHconn,         /* Address of connection handle */
PMQHOBJ  pHobj,          /* Address of object handle */
PMQLONG  pSelectorCount, /* Address of selector count */
PPMQLONG ppSelectors,    /* Address of ptr to array of selectors */
PMQLONG  pIntAttrCount;  /* Address of count of integer attributes */
PPMQLONG ppIntAttrs,     /* Address of ptr to array of integer attributes */
PMQLONG  pCharAttrLength, /* Address of character attribute length */
PPMQCHAR ppCharAttrs,    /* Address of ptr to character attributes array */
PMQLONG  pCompCode,      /* Address of completion code */
PMQLONG  pReason;        /* Address of reason code qualifying completion
                          code */
```

### ***Ouvrir-MQ\_OPEN\_EXIT***

MQ\_OPEN\_EXIT fournit une fonction d'exit ouvert permettant d'exécuter *avant* et *après* le traitement de l'appel MQOPEN. Utilisez l'identificateur de fonction MQXF\_OPEN avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer les fonctions d'exit d'appel MQOPEN *avant* et *après* .

L'interface de cette fonction est

```
MQ_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &Options,
              &pHobj, &CompCode, &Reason)
```

où les paramètres sont :

#### **ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

#### **ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

#### **Hconn (MQHCONN)-entrée**

Descripteur de connexion.

#### **pObjDescription (PMQOD)-entrée/sortie**

Pointeur vers le descripteur d'objet.

#### **Options (MQLONG)-entrée / sortie**

Options d'ouverture.

#### **pHobj (PMQHOBj)-entrée**

Pointeur vers le descripteur d'objet.

#### **CompCode (MQLONG)-entrée/sortie**

Code achèvement, valeurs valides pour lesquelles:

##### **MQCC\_OK**

Achèvement réussi.

##### **MQCC\_WARNING**

Achèvement partiel

##### **MQCC\_FAILED**

Echec de l'appel

#### **Motif (MQLONG)-entrée / sortie**

Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC\_OK, la seule valeur valide est:

## **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC\_FAILED ou MQCC\_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC\_\* valide.

## **Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP      ExitParms;      /* Exit parameter structure */
MQAXC      ExitContext;    /* Exit context structure */
MQHCONN    Hconn;         /* Connection handle */
PMQOD      pObjDesc;      /* Ptr to object descriptor */
MQLONG     Options;       /* Open options */
PMQHOBJS   pHobj;        /* Ptr to object handle */
MQLONG     CompCode;      /* Completion code */
MQLONG     Reason;       /* Reason code */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &Options,
              &pHobj, &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
void MQENTRY MQ_OPEN_EXIT (
  PMQAXP      pExitParms,    /* Address of exit parameter structure */
  PMQAXC      pExitContext,  /* Address of exit context structure */
  PMQHCONN    pHconn,       /* Address of connection handle */
  PPMQOD      ppObjDesc,     /* Address of ptr to object descriptor */
  PMQLONG     pOptions,      /* Address of open options */
  PPMQHOBJS   ppHobj,       /* Address of ptr to object handle */
  PMQLONG     pCompCode,     /* Address of completion code */
  PMQLONG     pReason);     /* Address of reason code qualifying
                             completion code */
```

## **Insertion-MQ\_PUT\_EXIT**

MQ\_PUT\_EXIT fournit une fonction d'exit d'insertion pour exécuter *avant* et *après* le traitement de l'appel MQPUT. Utilisez l'identificateur de fonction MQXF\_PUT avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer les fonctions d'exit d'appel MQPUT *avant* et *après* .

L'interface de cette fonction est:

```
MQ_PUT_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
            &pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)
```

où les paramètres sont :

### **ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

### **ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

### **Hconn (MQHCONN)-entrée**

Descripteur de connexion.

### **Hobj (MQHOBJS)-entrée/sortie**

Descripteur d'objet.

### **pMsgDescription (PMQMD)-entrée/sortie**

Pointeur vers le descripteur de message.

### **pPutMsgOpts (PMQPMO)-entrée/sortie**

Pointeur permettant d'insérer des options de message.

### **BufferLength (MQLONG)-entrée/sortie**

Longueur de la mémoire tampon de messages.

### **pBuffer (PMQBYTE)-entrée/sortie**

Pointeur vers la mémoire tampon de messages.

### **CompCode (MQLONG)-entrée/sortie**

Code achèvement, valeurs valides pour lesquelles:

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_WARNING**

Achèvement partiel.

#### **MQCC\_FAILED**

Echec de l'appel

### **Motif (MQLONG)-entrée / sortie**

Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC\_OK, la seule valeur valide est:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC\_FAILED ou MQCC\_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC\_ \* valide.

## **Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP      ExitParms;      /* Exit parameter structure */
MQAXC      ExitContext;    /* Exit context structure */
MQHCONN    Hconn;         /* Connection handle */
MQHOBJ     Hobj;         /* Object handle */
PMQMD      pMsgDesc;      /* Ptr to message descriptor */
PMQPMO     pPutMsgOpts;   /* Ptr to put message options */
MQLONG     BufferLength;   /* Message buffer length */
PMQBYTE    pBuffer;      /* Ptr to message data */
MQLONG     CompCode;      /* Completion code */
MQLONG     Reason;       /* Reason code */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_PUT_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
             &pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
void MQENTRY MQ_PUT_EXIT (
PMQAXP      pExitParms,    /* Address of exit parameter structure */
PMQAXC      pExitContext,  /* Address of exit context structure */
PMQHCONN    pHconn,       /* Address of connection handle */
PMQHOBJ     pHobj,        /* Address of object handle */
PPMQMD      ppMsgDesc,    /* Address of ptr to message descriptor */
PPMQPMO     ppPutMsgOpts, /* Address of ptr to put message options */
PMQLONG     pBufferLength, /* Address of message buffer length */
PPMQBYTE    ppBuffer,     /* Address of ptr to message buffer */
PMQLONG     pCompCode,    /* Address of completion code */
PMQLONG     pReason);     /* Address of reason code qualifying
                             completion code */
```

## **Remarques sur l'utilisation**

- Les messages de rapport générés par le gestionnaire de files d'attente ignorent le traitement normal des appels. Par conséquent, ces messages ne peuvent pas être interceptés par la fonction

MQ\_PUT\_EXIT ou la fonction MQPUT1 . Toutefois, les messages de rapport générés par l'agent MCA sont traités normalement et peuvent donc être interceptés par la fonction MQ\_PUT\_EXIT ou la fonction MQ\_PUT1\_EXIT . Pour être sûr d'intercepter tous les messages de rapport générés par l'agent MCA, vous devez utiliser à la fois MQ\_PUT\_EXIT et MQ\_PUT1\_EXIT .

### **Put1 - MQ\_PUT1\_EXIT**

MQ\_PUT1\_EXIT fournit une fonction d'exit *put one message only* pour exécuter *avant et après le traitement de l'appel* MQPUT1 . Utilisez l'identificateur de fonction MQXF\_PUT1 avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer *avant et après* les fonctions d'exit d'appel MQPUT1 .

L'interface de cette fonction est:

```
MQ_PUT1_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &pMsgDesc,  
&pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)
```

où les paramètres sont :

#### **ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

#### **ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

#### **Hconn (MQHCONN)-entrée**

Descripteur de connexion.

#### **pObjDescription (PMQOD)-entrée/sortie**

Pointeur vers le descripteur d'objet.

#### **pMsgDescription (PMQMD)-entrée/sortie**

Pointeur vers le descripteur de message.

#### **pPutMsgOpts (PMQPMO)-entrée/sortie**

Pointeur permettant d'insérer des options de message.

#### **BufferLength (MQLONG)-entrée/sortie**

Longueur de la mémoire tampon de messages.

#### **pBuffer (PMQBYTE)-entrée/sortie**

Pointeur vers la mémoire tampon de messages.

#### **CompCode (MQLONG)-entrée/sortie**

Code achèvement, valeurs valides pour lesquelles:

##### **MQCC\_OK**

Achèvement réussi.

##### **MQCC\_WARNING**

Achèvement partiel.

##### **MQCC\_FAILED**

Echec de l'appel

#### **Motif (MQLONG)-entrée / sortie**

Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC\_OK, la seule valeur valide est:

##### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC\_FAILED ou MQCC\_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC\_\* valide.

## **Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```

MQAXP      ExitParms;      /* Exit parameter structure */
MQAXC      ExitContext;  /* Exit context structure */
MQHCONN    Hconn;        /* Connection handle */
PMQOD      pObjDesc;     /* Ptr to object descriptor */
PMQMD      pMsgDesc;     /* Ptr to message descriptor */
PMQPMO     pPutMsgOpts;  /* Ptr to put message options */
MQLONG     BufferLength;  /* Message buffer length */
PMQBYTE    pBuffer;      /* Ptr to message data */
MQLONG     CompCode;     /* Completion code */
MQLONG     Reason;       /* Reason code */

```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```

MQ_PUT1_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &pMsgDesc,
              &pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)

```

Votre exit doit correspondre au prototype de fonction C suivant:

```

void MQENTRY MQ_PUT1_EXIT (
PMQAXP      pExitParms,      /* Address of exit parameter structure */
PMQAXC      pExitContext,    /* Address of exit context structure */
MQHCONN     pHconn,         /* Address of connection handle */
PMQOD      pObjDesc,        /* Address of ptr to object descriptor */
PPMQMD      ppMsgDesc,      /* Address of ptr to message descriptor */
PPMQPMO     ppPutMsgOpts,   /* Address of ptr to put message options */
MQLONG      pBufferLength,  /* Address of message buffer length */
PMQBYTE     pBuffer,        /* Address of ptr to message buffer */
MQLONG      pCompCode,      /* Address of completion code */
MQLONG      pReason);      /* Address of reason code qualifying
                             completion code */

```

### Définir-MQ\_SET\_EXIT

MQ\_SET\_EXIT fournit une fonction d'exit de définition permettant d'exécuter *avant* et *après le traitement des appels MQSET*. Utilisez l'identificateur de fonction MQXF\_SET avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer les fonctions d'exit d'appel MQSET *avant* et *après*.

L'interface de cette fonction est:

```

MQ_SET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
             &pCharAttr, &CompCode, &Reason)

```

où les paramètres sont :

#### ExitParms (MQAXP)-entrée/sortie

Structure des paramètres d'exit.

#### ExitContext (MQAXC)-entrée/sortie

Quittez la structure de contexte.

#### Hconn (MQHCONN)-entrée

Descripteur de connexion.

#### Hobj (MQHOBJ)-entrée

Descripteur d'objet.

#### SelectorCount (MQLONG)-entrée

Nombre de sélecteurs

#### pSelectors (PMQLONG)-entrée/sortie

Pointeur vers un tableau de valeurs de sélecteur.

#### IntAttrCount (MQLONG)-entrée

Nombre d'attributs de type entier.

#### Attributs pInt(PMQLONG)-entrée/sortie

Pointeur vers un tableau de valeurs d'attribut entières.

### CharAttrLongueur (MQLONG)-entrée/sortie

Longueur du tableau d'attributs de caractères.

### pCharAttr (PMQCHAR)-entrée / sortie

Pointeur vers les valeurs d'attribut de caractère.

### CompCode (MQLONG)-entrée/sortie

Code achèvement, valeurs valides pour lesquelles:

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_WARNING**

Achèvement partiel.

#### **MQCC\_FAILED**

Echec de l'appel

### Motif (MQLONG)-entrée / sortie

Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC\_OK, la seule valeur valide est:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC\_FAILED ou MQCC\_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC\_ \* valide.

## Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP    ExitParms;          /* Exit parameter structure */
MQAXC    ExitContext;       /* Exit context structure */
MQHCONN  Hconn;             /* Connection handle */
MQHOBJ   Hobj;              /* Object handle */
MQLONG   SelectorCount;     /* Count of selectors */
PMQLONG  pSelectors;        /* Ptr to array of attribute selectors */
MQLONG   IntAttrCount;      /* Count of integer attributes */
PMQLONG  pIntAttr;          /* Ptr to array of integer attributes */
MQLONG   CharAttrLength;    /* Length of char attributes array */
PMQCHAR  pCharAttr;         /* Ptr to character attributes */
MQLONG   CompCode;          /* Completion code */
MQLONG   Reason;           /* Reason code qualifying completion code */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_SET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttr, &CharAttrLength,
             &pCharAttr, &CompCode, &Reason)
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
void MQENTRY MQ_SET_EXIT (
PMQAXP    pExitParms,        /* Address of exit parameter structure */
PMQAXC    pExitContext,     /* Address of exit context structure */
PMQHCONN  pHconn,           /* Address of connection handle */
PMQHOBJ   pHobj,            /* Address of object handle */
PMQLONG   pSelectorCount,   /* Address of selector count */
PPMQLONG  ppSelectors,      /* Address of ptr to array of selectors */
PMQLONG   pIntAttrCount;    /* Address of count of integer attributes */
PPMQLONG  ppIntAttr,        /* Address of ptr to array of integer attributes */
PMQLONG   pCharAttrLength,  /* Address of character attribute length */
PPMQLONG  ppCharAttr,       /* Address of ptr to character attributes array */
PMQLONG   pCompCode,        /* Address of completion code */
PMQLONG   pReason);         /* Address of reason code qualifying completion
                             code */
```

## Statut-MQ\_STAT\_EXIT

MQ\_STAT\_EXIT fournit une fonction d'exit de statut permettant d'exécuter *avant* et *après* le traitement des appels MQSTAT. Utilisez l'identificateur de fonction MQXF\_STAT avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer les fonctions d'exit d'appel MQSTAT *avant* et *après* .

L'interface de cette fonction est:

```
MQ_STAT_EXIT (&ExitParms, &ExitContext, &Hconn, &Type, &pStatus  
             &CompCode, &Reason)
```

où les paramètres sont :

### ExitParms (MQAXP)-entrée/sortie

Structure des paramètres d'exit.

### ExitContext (MQAXC)-entrée/sortie

Quittez la structure de contexte.

### Hconn (MQHCONN)-entrée

Descripteur de connexion.

### Type (MQLONG)-entrée

Type d'informations de statut à extraire.

### pStatus (PMQSTS)-sortie

Pointeur vers la mémoire tampon d'état.

### CompCode (MQLONG)-entrée/sortie

Code achèvement, valeurs valides pour lesquelles:

#### MQCC\_OK

Achèvement réussi.

#### MQCC\_WARNING

Achèvement partiel.

#### MQCC\_FAILED

Echec de l'appel

### Motif (MQLONG)-entrée / sortie

Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC\_OK, la seule valeur valide est:

#### MQRC\_NONE

(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC\_FAILED ou MQCC\_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC\_\* valide.

## Appel du langage C

Votre exit doit correspondre au prototype de fonction C suivant:

```
void MQENTRY MQ_STAT_EXIT (  
PMQAXP  pExitParms,      /* Address of exit parameter structure */  
PMQAXC  pExitContext,   /* Address of exit context structure */  
PMQHCONN pHconn,        /* Address of connection handle */  
PMQLONG pType            /* Address of status type */  
PPMQSTS ppStatus        /* Address of status buffer */  
PMQLONG pCompCode,      /* Address of completion code */  
PMQLONG pReason);       /* Address of reason code qualifying completion  
                           code */
```

## Arrêt-MQ\_TERM\_EXIT

MQ\_TERM\_EXIT fournit un arrêt au niveau de la connexion, enregistré avec l'identificateur de fonction MQXF\_TERM et ExitReason MQXR\_CONNECTION. S'il est enregistré, MQ\_TERM\_EXIT est appelé une fois pour chaque demande de déconnexion.

Dans le cadre de l'arrêt, le stockage qui n'est plus requis par l'exit peut être libéré et tout nettoyage requis peut être effectué.

Si un MQ\_TERM\_EXIT échoue avec MQXCC\_FAILED, le gestionnaire de files d'attente est renvoyé par le MQDISC qui l'a appelé avec MQCC\_FAILED et MQRC\_API\_EXIT\_ERROR.

Si le gestionnaire de files d'attente rencontre une erreur lors de l'arrêt de l'environnement d'exécution de la fonction d'exit d'API après avoir appelé le dernier MQ\_TERM\_EXIT, le gestionnaire de files d'attente renvoie l'appel MQDISC qui a appelé MQ\_TERM\_EXIT avec MQCC\_FAILED et MQRC\_API\_EXIT\_TERM\_ERROR

L'interface de cette fonction est:

```
MQ_TERM_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

où les paramètres sont :

#### **ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

#### **ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

#### **CompCode (MQLONG)-entrée/sortie**

Code achèvement, valeurs valides pour lesquelles:

##### **MQCC\_OK**

Achèvement réussi.

##### **MQCC\_FAILED**

Echec de l'appel

#### **Motif (MQLONG)-entrée / sortie**

Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC\_OK, la seule valeur valide est:

##### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC\_FAILED, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC\_\* valide.

Le CompCode et le motif renvoyés à l'application dépendent de la valeur de la zone ExitResponse dans MQAXP.

## **Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP      ExitParms;      /* Exit parameter structure */
MQAXC      ExitContext;    /* Exit context structure */
MQLONG     CompCode;      /* Completion code */
MQLONG     Reason;        /* Reason code */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_TERM_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
void MQENTRY MQ_TERM_EXIT (
PMQAXP      pExitParms,    /* Address of exit parameter structure */
PMQAXC      pExitContext,  /* Address of exit context structure */
PMQLONG     pCompCode,     /* Address of completion code */
```



```
PMQLONG      pReason);      /* Address of reason code qualifying
                             completion code */
```

## Remarques sur l'utilisation

1. La fonction MQ\_TERM\_EXIT est facultative. Il n'est pas nécessaire qu'une suite d'exit enregistre un exit d'arrêt s'il n'y a pas de traitement d'arrêt à effectuer.

Si des fonctions appartenant à la suite d'exit acquièrent des ressources lors de la connexion, une fonction MQ\_TERM\_EXIT est un point pratique pour libérer ces ressources, par exemple pour libérer de la mémoire obtenue de manière dynamique.

2. Si une fonction MQ\_TERM\_EXIT est enregistrée lors de l'émission de l'appel MQDISC, la fonction d'exit est appelée une fois que toutes les fonctions d'exit MQDISC ont été appelées.
3. Si MQ\_TERM\_EXIT renvoie MQXCC\_FAILED dans la zone ExitResponse de MQAXP, ou échoue d'une autre manière, l'appel MQDISC qui a provoqué l'appel de MQ\_TERM\_EXIT échoue également, avec les paramètres **CompCode** et **Reason** définis sur les valeurs appropriées.

## Enregistrement de l'abonnement-MQ\_SUB\_EXIT

MQ\_SUB\_EXIT fournit une fonction d'exit permettant d'exécuter *avant* et *après le traitement de réenregistrement d'abonnement*. Utilisez l'identificateur de fonction MQXF\_SUB avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer *avant* et *après* les fonctions d'exit d'appel d'enregistrement d'abonnement.

L'interface de cette fonction est:

```
MQ_SUB_EXIT (&ExitParms, &ExitContext, &Hconn, &pSubDesc, &pHobj, &pHsub, &CompCode, &Reason)
```

où les paramètres sont :

### ExitParms (MQAXP)-entrée/sortie

Structure des paramètres d'exit.

### ExitContext (MQAXC)-entrée/sortie

Quittez la structure de contexte.

### Hconn (MQHCONN)-entrée/sortie

Descripteur de connexion.

### pSubDesc-input/output

Tableau de sélecteurs d'attribut.

### pHobj -entrée/sortie

Descripteur d'objet

### Entrée / sortie pHsub (MQHOBJ)

Descripteur d'abonnement

### CompCode (MQLONG)-entrée/sortie

Code achèvement, valeurs valides pour lesquelles:

#### MQCC\_OK

Achèvement réussi.

#### MQCC\_WARNING

Achèvement partiel.

#### MQCC\_FAILED

Echec de l'appel

### Motif (MQLONG)-entrée / sortie

Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC\_OK, la seule valeur valide est:

#### MQRC\_NONE

(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC\_FAILED ou MQCC\_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC\_\* valide.

## Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP    ExitParms;    /* Exit parameter structure */
MQAXC    ExitContext;  /* Exit context structure */
MQHCONN  Hconn;       /* Connection handle */
PMQSD    pSubDesc;    /* Subscription descriptor */
PMQHOBJS pHobj;       /* Object Handle */
PMQHOBJS pHsub;       /* Subscription handle */
MQLONG   CompCode;    /* Completion code */
MQLONG   Reason;      /* Reason code qualifying completion code */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_SUB_EXIT (&ExitParms, &ExitContext, &Hconn, &pSubDesc, &pHobj, &pHsub,
             &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
PMQAXP    pExitParms;    /* Exit parameter structure */
PMQAXC    pExitContext;  /* Exit context structure */
PMQHCONN  pHconn;       /* Connection handle */
PPMQSD    ppSubDesc;    /* Subscription descriptor */
PPMQHOBJS ppHobj;       /* Object Handle */
PPMQHOBJS ppHsub;       /* Subscription handle */
PMQLONG   pCompCode;    /* Completion code */
PMQLONG   pReason;      /* Reason code qualifying completion code */
```

### ***Demande d'abonnement-MQ\_SUBRQ\_EXIT***

MQ\_SUBRQ\_EXIT fournit une fonction d'exit de demande d'abonnement permettant d'exécuter *avant* et *après* le traitement de la demande d'abonnement. Utilisez l'identificateur de fonction MQXF\_SUBRQ avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer *avant* et *après* les fonctions d'exit d'appel de demande d'abonnement.

L'interface de cette fonction est:

```
MQ_SUBRQ_EXIT (&ExitParms, &ExitContext, &Hconn, &pHsub, &Action, &pSubRqOpts,
              &CompCode, &Reason)
```

où les paramètres sont :

#### **ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

#### **ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

#### **Hconn (MQHCONN)-entrée/sortie**

Descripteur de connexion.

#### **Entrée / sortie pHsub (MQHOBJS)**

Descripteur d'abonnement

#### **Entrée / sortie d'action (MQLONG)**

Action

#### **Entrée / sortie pSubRqOpts (MQSRO)**

#### **CompCode (MQLONG)-entrée/sortie**

Code achèvement, valeurs valides pour lesquelles:

**MQCC\_OK**

Achèvement réussi.

**MQCC\_WARNING**

Achèvement partiel.

**MQCC\_FAILED**

Echec de l'appel

**Motif (MQLONG)-entrée / sortie**

Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC\_OK, la seule valeur valide est:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC\_FAILED ou MQCC\_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC\_\* valide.

**Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP    ExitParms;      /* Exit parameter structure */
MQAXC    ExitContext;   /* Exit context structure */
MQHCONN  Hconn;         /* Connection handle */
PMQLONG  pHsub;         /* Subscription handle */
MQLONG   Action;        /* Action */
PMQSRO   pSubRqOpts;   /* Subscription Request Options */
MQLONG   CompCode;     /* Completion code */
MQLONG   Reason;       /* Reason code qualifying completion code */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_SUBRQ_EXIT (&ExitParms, &ExitContext, &Hconn, &pHsub, &Action, &pSubRqOpts,
               &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
void MQENTRY MQ_SUBRQ_EXIT (
PMQAXP    pExitParms,      /* Address of exit parameter structure */
PMQAXC    pExitContext,   /* Address of exit context structure */
PMQHCONN  pHconn,         /* Address of connection handle */
PPMQHOBJS ppHsub;        /* Address of Subscription handle */
PMQLONG   pAction;        /* Address of Action */
PPMQSRO   ppSubRqOpts;    /* Address of Subscription Request Options */
PMQLONG   pCompCode,     /* Address of completion code */
PMQLONG   pReason;       /* Address of reason code qualifying completion
                           code */
```

**xa\_close-XA\_CLOSE\_EXIT**

XA\_CLOSE\_EXIT fournit une fonction d'exit xa\_close à exécuter avant et après le traitement de xa\_close. Utilisez l'identificateur de fonction MQXF\_XACLOSE avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer les fonctions d'exit d'appel xa\_close avant et après.

L'interface de cette fonction est:

```
XA_CLOSE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

**ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

**ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

**Hconn (MQHCONN)-entrée**

Descripteur de connexion.

**pXa\_info (PMQCHAR)-entrée/sortie**

Informations sur le gestionnaire de ressources spécifique à l'instance.

**Rmid (MQLONG)-entrée / sortie**

Identificateur du gestionnaire de ressources.

**Indicateurs (MQLONG)-entrée/sortie**

Options du gestionnaire de ressources.

**XARetCode (MQLONG)-entrée/sortie**

Réponse de l'appel XA.

**Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP    ExitParms;    /* Exit parameter structure */
MQAXC    ExitContext; /* Exit context structure */
MQHCONN  Hconn;       /* Connection handle */
PMQCHAR  pXa_info;   /* Instance-specific RM info */
MQLONG   Rmid;       /* Resource manager identifier */
MQLONG   Flags;      /* Resource manager options*/
MQLONG   XARetCode; /* Response from XA call */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_CLOSE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
typedef void MQENTRY XA_CLOSE_EXIT (
    PMQAXP    pExitParms, /* Address of exit parameter structure */
    PMQAXC    pExitContext, /* Address of exit context structure */
    PMQHCONN  pHconn,      /* Address of connection handle */
    PPMQCHAR  ppXa_info,   /* Address of instance-specific RM info */
    PMQLONG   pRmid,       /* Address of resource manager identifier */
    PMQLONG   pFlags,      /* Address of resource manager options*/
    PMQLONG   pXARetCode); /* Address of response from XA call */
```

***xa\_commit-XA\_COMMIT\_EXIT***

XA\_COMMIT\_EXIT fournit une fonction d'exit xa\_commit à exécuter avant et après le traitement de xa\_commit. Utilisez l'identificateur de fonction MQXF\_XACOMMIT avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer les fonctions d'exit d'appel avant et après xa\_commit.

L'interface de cette fonction est:

```
XA_COMMIT_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

**ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

**ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

**Hconn (MQHCONN)-entrée**

Descripteur de connexion.

**pXID (MQPTR)-entrée/sortie**

ID de branche de transaction.

**Rmid (MQLONG)-entrée / sortie**

Identificateur du gestionnaire de ressources.

**Indicateurs (MQLONG)-entrée/sortie**

Options du gestionnaire de ressources.

**XARetCode (MQLONG)-entrée/sortie**

Réponse de l'appel XA.

**Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP    ExitParms;    /* Exit parameter structure */
MQAXC    ExitContext; /* Exit context structure */
MQHCONN  Hconn;       /* Connection handle */
MQPTR    pXID;        /* Transaction branch ID */
MQLONG   Rmid;        /* Resource manager identifier */
MQLONG   Flags;       /* Resource manager options*/
MQLONG   XARetCode;  /* Response from XA call */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_COMMIT_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
typedef void MQENTRY XA_COMMIT_EXIT (
    PMQAXP    pExitParms, /* Address of exit parameter structure */
    PMQAXC    pExitContext, /* Address of exit context structure */
    PMQHCONN  pHconn, /* Address of connection handle */
    PMQPTR    ppXID, /* Address of transaction branch ID */
    PMQLONG   pRmid, /* Address of resource manager identifier */
    PMQLONG   pFlags, /* Address of resource manager options*/
    PMQLONG   pXARetCode); /* Address of response from XA call */
```

***xa\_complete-XA\_COMPLETE\_EXIT***

XA\_COMPLETE\_EXIT fournit une fonction d'exit *xa\_complete* à exécuter avant et après le traitement *xa\_complete*. Utilisez l'identificateur de fonction MQXF\_XACOMPLETE avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer les fonctions d'exit d'appel avant et après *xa\_complete*.

L'interface de cette fonction est:

```
XA_COMPLETE_EXIT (&ExitParms, &ExitContext, &Hconn, &pHandle, &pRetVal, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

**ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

**ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

**Hconn (MQHCONN)-entrée**

Descripteur de connexion.

**pHandle (PMQLONG)-entrée/sortie**

Pointeur vers l'opération asynchrone.

**pRetVal (PMQLONG)-entrée/sortie**

Valeur de retour de l'opération asynchrone.

### **Rmid (MQLONG)-entrée / sortie**

Identificateur du gestionnaire de ressources.

### **Indicateurs (MQLONG)-entrée/sortie**

Options du gestionnaire de ressources.

### **XARetCode (MQLONG)-entrée/sortie**

Réponse de l'appel XA.

## **Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP  ExitParms;    /* Exit parameter structure */
MQAXC  ExitContext; /* Exit context structure */
MQHCONN Hconn;      /* Connection handle */
PMQLONG pHandle;    /* Ptr to asynchronous op */
PMQLONG pRetval;    /* Return value of async op */
MQLONG  Rmid;       /* Resource manager identifier */
MQLONG  Flags;      /* Resource manager options*/
MQLONG  XARetCode; /* Response from XA call */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_COMPLETE_EXIT (&ExitParms, &ExitContext, &Hconn, &pHandle, &pRetval, &Rmid, &Flags,
&XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
typedef void MQENTRY XA_COMPLETE_EXIT (
    PMQAXP  pExitParms, /* Address of exit parameter structure */
    PMQAXC  pExitContext, /* Address of exit context structure */
    PMQHCONN pHconn, /* Address of connection handle */
    PPMQLONG ppHandle, /* Address of ptr to asynchronous op */
    PPMQLONG ppRetval, /* Address of return value of async op */
    PMQLONG pRmid, /* Address of resource manager identifier */
    PMQLONG pFlags, /* Address of resource manager options*/
    PMQLONG pXARetCode); /* Address of response from XA call */
```

### ***xa\_end-XA\_END\_EXIT***

XA\_END\_EXIT fournit une fonction d'exit `xa_end` à exécuter avant et après le traitement `xa_end`. Utilisez l'identificateur de fonction `MQXF_XAEND` avec les raisons d'exit `MQXR_BEFORE` et `MQXR_AFTER` pour enregistrer les fonctions d'exit d'appel avant et après `xa_end`.

L'interface de cette fonction est:

```
XA_END_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

### **ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

### **ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

### **Hconn (MQHCONN)-entrée**

Descripteur de connexion.

### **pXID (MQPTR)-entrée/sortie**

ID de branche de transaction.

### **Rmid (MQLONG)-entrée / sortie**

Identificateur du gestionnaire de ressources.

### **Indicateurs (MQLONG)-entrée/sortie**

Options du gestionnaire de ressources.

## **XARetCode (MQLONG)-entrée/sortie**

Réponse de l'appel XA.

## **Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP  ExitParms;    /* Exit parameter structure */
MQAXC  ExitContext; /* Exit context structure */
MQHCONN Hconn;      /* Connection handle */
MQPTR  pXID;        /* Transaction branch ID */
MQLONG Rmid;        /* Resource manager identifier */
MQLONG Flags;       /* Resource manager options*/
MQLONG XARetCode;   /* Response from XA call */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_END_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
typedef void MQENTRY XA_END_EXIT (
    PMQAXP  pExitParms,    /* Address of exit parameter structure */
    PMQAXC  pExitContext, /* Address of exit context structure */
    PMQHCONN pHconn,      /* Address of connection handle */
    PMQPTR  ppXID,        /* Address of transaction branch ID */
    PMQLONG pRmid,        /* Address of resource manager identifier */
    PMQLONG pFlags,       /* Address of resource manager options*/
    PMQLONG pXARetCode); /* Address of response from XA call */
```

## ***xa\_forget-XA\_FORGET\_EXIT***

XA\_FORGET\_EXIT fournit une fonction d'exit `xa_forget` à exécuter avant et après le traitement de `xa_forget`. Utilisez l'identificateur de fonction `MQXF_XAFORGET` avec les raisons d'exit `MQXR_BEFORE` et `MQXR_AFTER` pour enregistrer les fonctions d'exit d'appel `xa_forget` avant et après.

L'interface de cette fonction est:

```
XA_FORGET_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

### **ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

### **ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

### **Hconn (MQHCONN)-entrée**

Descripteur de connexion.

### **pXID (MQPTR)-entrée/sortie**

ID de branche de transaction.

### **Rmid (MQLONG)-entrée / sortie**

Identificateur du gestionnaire de ressources.

### **Indicateurs (MQLONG)-entrée/sortie**

Options du gestionnaire de ressources.

### **XARetCode (MQLONG)-entrée/sortie**

Réponse de l'appel XA.

## **Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```

MQAXP  ExitParms;    /* Exit parameter structure */
MQAXC  ExitContext; /* Exit context structure */
MQHCONN Hconn;      /* Connection handle */
MQPTR  pXID;        /* Transaction branch ID */
MQLONG Rmid;        /* Resource manager identifier */
MQLONG Flags;       /* Resource manager options*/
MQLONG XARetCode;  /* Response from XA call */

```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_FORGET_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```

typedef void MQENTRY XA_FORGET_EXIT (
    PMQAXP  pExitParms, /* Address of exit parameter structure */
    PMQAXC  pExitContext, /* Address of exit context structure */
    PMQHCONN pHconn, /* Address of connection handle */
    PMQPTR  ppXID, /* Address of transaction branch ID */
    PMQLONG pRmid, /* Address of resource manager identifier */
    PMQLONG pFlags, /* Address of resource manager options*/
    PMQLONG pXARetCode); /* Address of response from XA call */

```

### ***xa\_open-XA\_OPEN\_EXIT***

XA\_OPEN\_EXIT fournit une fonction d'exit xa\_open à exécuter avant et après le traitement xa\_open. Utilisez l'identificateur de fonction MQXF\_XAOPEN avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer les fonctions d'exit d'appel xa\_open avant et après.

L'interface de cette fonction est:

```
XA_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

#### **ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

#### **ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

#### **Hconn (MQHCONN)-entrée**

Descripteur de connexion.

#### **pXa\_info (PMQCHAR)-entrée/sortie**

Informations sur le gestionnaire de ressources spécifique à l'instance.

#### **Rmid (MQLONG)-entrée / sortie**

Identificateur du gestionnaire de ressources.

#### **Indicateurs (MQLONG)-entrée/sortie**

Options du gestionnaire de ressources.

#### **XARetCode (MQLONG)-entrée/sortie**

Réponse de l'appel XA.

### **Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```

MQAXP  ExitParms;    /* Exit parameter structure */
MQAXC  ExitContext; /* Exit context structure */
MQHCONN Hconn;      /* Connection handle */
PMQCHAR pXa_info;   /* Instance-specific RM info */
MQLONG Rmid;        /* Resource manager identifier */

```



```
MQLONG Flags; /* Resource manager options*/
MQLONG XARetCode; /* Response from XA call */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
typedef void MQENTRY XA_OPEN_EXIT (
    PMQAXP pExitParms, /* Address of exit parameter structure */
    PMQAXC pExitContext, /* Address of exit context structure */
    PMQHCONN pHconn, /* Address of connection handle */
    PPMQCHAR ppXa_info, /* Address of instance-specific RM info */
    PMQLONG pRmid, /* Address of resource manager identifier */
    PMQLONG pFlags, /* Address of resource manager options*/
    PMQLONG pXARetCode); /* Address of response from XA call */
```

### ***xa\_prepare-XA\_PREPARE\_EXIT***

XA\_PREPARE\_EXIT fournit une fonction d'exit `xa_prepare` à exécuter avant et après le traitement `xa_prepare`. Utilisez l'identificateur de fonction `MQXF_XAPREPARE` avec les raisons d'exit `MQXR_BEFORE` et `MQXR_AFTER` pour enregistrer les fonctions d'exit d'appel `xa_prepare` avant et après.

L'interface de cette fonction est:

```
XA_PREPARE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

#### **ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

#### **ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

#### **Hconn (MQHCONN)-entrée**

Descripteur de connexion.

#### **pXID (MQPTR)-entrée/sortie**

ID de branche de transaction.

#### **Rmid (MQLONG)-entrée / sortie**

Identificateur du gestionnaire de ressources.

#### **Indicateurs (MQLONG)-entrée/sortie**

Options du gestionnaire de ressources.

#### **XARetCode (MQLONG)-entrée/sortie**

Réponse de l'appel XA.

### **Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP ExitParms; /* Exit parameter structure */
MQAXC ExitContext; /* Exit context structure */
MQHCONN Hconn; /* Connection handle */
MQPTR pXID; /* Transaction branch ID */
MQLONG Rmid; /* Resource manager identifier */
MQLONG Flags; /* Resource manager options*/
MQLONG XARetCode; /* Response from XA call */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_PREPARE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
typedef void MQENTRY XA_PREPARE_EXIT (  
    PMQAXP    pExitParms, /* Address of exit parameter structure */  
    PMQAXC    pExitContext, /* Address of exit context structure */  
    PMQHCONN  pHconn, /* Address of connection handle */  
    PMQPTR    ppXID, /* Address of transaction branch ID */  
    MQLONG    pRmid, /* Address of resource manager identifier */  
    MQLONG    pFlags, /* Address of resource manager options*/  
    MQLONG    pXARetCode); /* Address of response from XA call */
```

### ***xa\_recover-XA\_RECOVER\_EXIT***

XA\_RECOVER\_EXIT fournit une fonction d'exit `xa_recover` à exécuter avant et après le traitement `xa_recover`. Utilisez l'identificateur de fonction `MQXF_XARECOVER` avec les raisons d'exit `MQXR_BEFORE` et `MQXR_AFTER` pour enregistrer les fonctions d'exit d'appel `xa_recover` avant et après.

L'interface de cette fonction est:

```
XA_RECOVER_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Count, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

#### **ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

#### **ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

#### **Hconn (MQHCONN)-entrée**

Descripteur de connexion.

#### **pXID (MQPTR)-entrée/sortie**

ID de branche de transaction.

#### **Nombre (MQLONG)-entrée / sortie**

Nombre maximal de XID dans le tableau XID

#### **Rmid (MQLONG)-entrée / sortie**

Identificateur du gestionnaire de ressources.

#### **Indicateurs (MQLONG)-entrée/sortie**

Options du gestionnaire de ressources.

#### **XARetCode (MQLONG)-entrée/sortie**

Réponse de l'appel XA.

### **Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP    ExitParms; /* Exit parameter structure */  
MQAXC    ExitContext; /* Exit context structure */  
MQHCONN  Hconn; /* Connection handle */  
MQPTR    pXID; /* Transaction branch ID */  
MQLONG    Count; /* Max XIDs in XID array */  
MQLONG    Rmid; /* Resource manager identifier */  
MQLONG    Flags; /* Resource manager options*/  
MQLONG    XARetCode; /* Response from XA call */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_RECOVER_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Count, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
typedef void MQENTRY XA_RECOVER_EXIT (  

```

```

PMQAXP  pExitParms, /* Address of exit parameter structure */
PMQAXC  pExitContext, /* Address of exit context structure */
PMQHCONN pHconn, /* Address of connection handle */
PMQPTR  ppXID, /* Address of transaction branch ID */
PMQLONG pCount, /* Address of max XIDs in XID array */
PMQLONG pRmid, /* Address of resource manager identifier */
PMQLONG pFlags, /* Address of resource manager options*/
PMQLONG pXARetCode); /* Address of response from XA call */

```

## **xa\_rollback-XA\_ROLLBACK\_EXIT**

XA\_ROLLBACK\_EXIT fournit une fonction d'exit xa\_rollback à exécuter avant et après le traitement de xa\_rollback. Utilisez l'identificateur de fonction MQXF\_XAROLLBACK avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer les fonctions d'exit d'appel xa\_rollback avant et après.

L'interface de cette fonction est:

```
XA_ROLLBACK_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

### **ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

### **ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

### **Hconn (MQHCONN)-entrée**

Descripteur de connexion.

### **pXID (MQPTR)-entrée/sortie**

ID de branche de transaction.

### **Rmid (MQLONG)-entrée / sortie**

Identificateur du gestionnaire de ressources.

### **Indicateurs (MQLONG)-entrée/sortie**

Options du gestionnaire de ressources.

### **XARetCode (MQLONG)-entrée/sortie**

Réponse de l'appel XA.

## **Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```

MQAXP  ExitParms; /* Exit parameter structure */
MQAXC  ExitContext; /* Exit context structure */
MQHCONN Hconn; /* Connection handle */
MQPTR  pXID; /* Transaction branch ID */
MQLONG Rmid; /* Resource manager identifier */
MQLONG Flags; /* Resource manager options*/
MQLONG XARetCode; /* Response from XA call */

```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_ROLLBACK_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```

typedef void MQENTRY XA_ROLLBACK_EXIT (
  PMQAXP  pExitParms, /* Address of exit parameter structure */
  PMQAXC  pExitContext, /* Address of exit context structure */
  PMQHCONN pHconn, /* Address of connection handle */
  PMQPTR  ppXID, /* Address of transaction branch ID */
  PMQLONG pRmid, /* Address of resource manager identifier */
  PMQLONG pFlags, /* Address of resource manager options*/
  PMQLONG pXARetCode); /* Address of response from XA call */

```

## ***xa\_start-XA\_START\_EXIT***

XA\_START\_EXIT fournit une fonction d'exit `xa_start` à exécuter avant et après le traitement `xa_start`. Utilisez l'identificateur de fonction `MQXF_XASTART` avec les raisons d'exit `MQXR_BEFORE` et `MQXR_AFTER` pour enregistrer les fonctions d'exit d'appel avant et après `xa_start`.

L'interface de cette fonction est:

```
XA_START_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

### **ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

### **ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

### **Hconn (MQHCONN)-entrée**

Descripteur de connexion.

### **pXID (MQPTR)-entrée/sortie**

ID de branche de transaction.

### **Rmid (MQLONG)-entrée / sortie**

Identificateur du gestionnaire de ressources.

### **Indicateurs (MQLONG)-entrée/sortie**

Options du gestionnaire de ressources.

### **XARetCode (MQLONG)-entrée/sortie**

Réponse de l'appel XA.

## **Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP  ExitParms; /* Exit parameter structure */
MQAXC  ExitContext; /* Exit context structure */
MQHCONN Hconn; /* Connection handle */
MQPTR  pXID; /* Transaction branch ID */
MQLONG Rmid; /* Resource manager identifier */
MQLONG Flags; /* Resource manager options*/
MQLONG XARetCode; /* Response from XA call */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_START_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
typedef void MQENTRY XA_START_EXIT (
    PMQAXP  pExitParms, /* Address of exit parameter structure */
    PMQAXC  pExitContext, /* Address of exit context structure */
    PMQHCONN pHconn, /* Address of connection handle */
    MQPTR  ppXID, /* Address of transaction branch ID */
    PMQLONG pRmid, /* Address of resource manager identifier */
    PMQLONG pFlags, /* Address of resource manager options*/
    PMQLONG pXARetCode); /* Address of response from XA call */
```

## ***ax\_reg-AX\_REG\_EXIT***

AX\_REG\_EXIT fournit une fonction d'exit `ax_reg` à exécuter avant et après le traitement `ax_reg`. Utilisez l'identificateur de fonction `MQXF_AXREG` avec les raisons d'exit `MQXR_BEFORE` et `MQXR_AFTER` pour enregistrer les fonctions d'exit d'appel `ax_reg` avant et après.

L'interface de cette fonction est:

```
AX_REG_EXIT (&ExitParms, &ExitContext, &pXID, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

**ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

**ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

**Hconn (MQHCONN)-entrée**

Descripteur de connexion.

**pXID (MQPTR)-entrée/sortie**

ID de branche de transaction.

**Rmid (MQLONG)-entrée / sortie**

Identificateur du gestionnaire de ressources.

**Indicateurs (MQLONG)-entrée/sortie**

Options du gestionnaire de ressources.

**XARetCode (MQLONG)-entrée/sortie**

Réponse de l'appel XA.

## Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP  ExitParms; /* Exit parameter structure */
MQAXC  ExitContext; /* Exit context structure */
MQPTR  pXID; /* Transaction branch ID */
MQLONG Rmid; /* Resource manager identifier */
MQLONG Flags; /* Resource manager options*/
MQLONG XARetCode; /* Response from XA call */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
AX_REG_EXIT (&ExitParms, &ExitContext, &pXID, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
typedef void MQENTRY AX_REG_EXIT (
    PMQAXP pExitParms, /* Address of exit parameter structure */
    PMQAXC pExitContext, /* Address of exit context structure */
    PMQPTR ppXID, /* Address of transaction branch ID */
    PMQLONG pRmid, /* Address of resource manager identifier */
    PMQLONG pFlags, /* Address of resource manager options*/
    PMQLONG pXARetCode); /* Address of response from XA call */
```

### **ax\_unreg-AX\_UNREG\_EXIT**

AX\_UNREG\_EXIT fournit une fonction d'exit ax\_unreg à exécuter avant et après le traitement ax\_unreg. Utilisez l'identificateur de fonction MQXF\_AXUNREG avec les raisons d'exit MQXR\_BEFORE et MQXR\_AFTER pour enregistrer les fonctions d'exit d'appel ax\_unreg avant et après.

L'interface de cette fonction est:

```
AX_UNREG_EXIT (&ExitParms, &ExitContext, &Rmid, &Flags, &XARetCode);
```

où les paramètres sont :

**ExitParms (MQAXP)-entrée/sortie**

Structure des paramètres d'exit.

**ExitContext (MQAXC)-entrée/sortie**

Quittez la structure de contexte.

**Rmid (MQLONG)-entrée / sortie**

Identificateur du gestionnaire de ressources.

**Indicateurs (MQLONG)-entrée/sortie**

Options du gestionnaire de ressources.

**XARetCode (MQLONG)-entrée/sortie**

Réponse de l'appel XA.

**Appel du langage C**

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP  ExitParms;    /* Exit parameter structure */
MQAXC  ExitContext; /* Exit context structure */
MQLONG Rmid;        /* Resource manager identifier */
MQLONG Flags;       /* Resource manager options*/
MQLONG XARetCode;   /* Response from XA call */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
AX_UNREG_EXIT (&ExitParms, &ExitContext, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
typedef void MQENTRY AX_UNREG_EXIT (
    PMQAXP  pExitParms, /* Address of exit parameter structure */
    PMQAXC  pExitContext, /* Address of exit context structure */
    PMQLONG pRmid,        /* Address of resource manager identifier */
    PMQLONG pFlags,       /* Address of resource manager options*/
    PMQLONG pXARetCode); /* Address of response from XA call */
```

**Informations générales sur l'appel des fonctions d'exit**

Cette rubrique fournit des conseils généraux pour vous aider à planifier vos exits, en particulier en ce qui concerne le traitement des erreurs et des événements inattendus.

**Incident de sortie**

Si une fonction d'exit s'arrête de manière anormale après un appel MQGET destructif, hors point de synchronisation, mais avant que le message n'ait été transmis à l'application, le gestionnaire d'exit peut récupérer de l'échec et transmettre le contrôle à l'application.

Dans ce cas, le message peut être perdu. Il s'agit de ce qui se produit lorsqu'une application échoue immédiatement après avoir reçu un message d'une file d'attente.

L'appel MQGET peut aboutir avec MQCC\_FAILED et MQRC\_API\_EXIT\_ERROR.

Si une fonction d'exit d'appel d'API *avant* s'arrête de manière anormale, le gestionnaire d'exit peut effectuer une reprise après l'échec et transmettre le contrôle à l'application sans traiter l'appel d'API. Dans ce cas, la fonction d'exit doit récupérer les ressources qu'elle possède.

Si des exits chaînés sont utilisés, les exits d'appel d'API *après* pour les exits d'appel d'API *avant* qui ont été déclenchés avec succès peuvent eux-mêmes être déclenchés. L'appel API peut échouer avec MQCC\_FAILED et MQRC\_API\_EXIT\_ERROR.

*Exemple de traitement d'erreurs pour les fonctions de sortie*

Le diagramme suivant montre les points (e N) où des erreurs peuvent se produire. Il ne s'agit que d'un exemple montrant comment les exits se comportent et doivent être lus avec le tableau suivant. Dans

cet exemple, deux fonctions d'exit sont appelées avant et après chaque appel d'API pour afficher le comportement avec des exits chaînés.

Application	ErrPt	Exit function	API call
Start			
MQCONN	-->		
	e1	MQ_INIT_EXIT	
	e2	before MQ_CONNX_EXIT 1	
	e3	before MQ_CONNX_EXIT 2	
	e4		--> MQCONN
	e5	after MQ_CONNX_EXIT 2	
	e6	after MQ_CONNX_EXIT 1	
	e7		
MQOPEN	<-- -->		
	e8	before MQ_OPEN_EXIT 1	
	e9	before MQ_OPEN_EXIT 2	
	e10		--> MQOPEN
	e11	after MQ_OPEN_EXIT 2	
	e12	after MQ_OPEN_EXIT 1	
MQPUT	<-- -->		
	e13	before MQ_PUT_EXIT 1	
	e14	before MQ_PUT_EXIT 2	
	e15		--> MQPUT
	e16	after MQ_PUT_EXIT 2	
	e17	after MQ_PUT_EXIT 1	
MQCLOSE	<-- -->		
	e18	before MQ_CLOSE_EXIT 1	
	e19	before MQ_CLOSE_EXIT 2	
	e20		--> MQCLOSE
	e21	after MQ_CLOSE_EXIT 2	
	e22	after MQ_CLOSE_EXIT 1	
MQDISC	<-- -->		
	e23	before MQ_DISC_EXIT 1	
	e24	before MQ_DISC_EXIT 2	
	e25		--> MQDISC
	e26	after MQ_DISC_EXIT 2	
	e27	after MQ_DISC_EXIT 1	
	<--		
end			

Le tableau suivant répertorie les actions à effectuer à chaque point d'erreur. Seul un sous-ensemble des points d'erreur a été couvert, car les règles affichées ici peuvent s'appliquer à tous les autres. Il s'agit des actions qui spécifient le comportement prévu dans chaque cas.

Tableau 837. Erreurs de sortie d'API et actions appropriées à effectuer

Err Pt	Description	Actions
e1	Erreur lors de la configuration de l'environnement.	<ol style="list-style-type: none"> <li>1. Annuler la configuration de l'environnement si nécessaire</li> <li>2. Aucune fonction d'exit de l'unité</li> <li>3. Echec de MQCONN avec MQCC_FAILED, MQRC_API_EXIT_LOAD_ERROR</li> </ol>
e2	La fonction MQ_INIT_EXIT se termine par: <ul style="list-style-type: none"> <li>• MQXCC_ECHEC</li> <li>• MQXCC_*</li> </ul>	<ul style="list-style-type: none"> <li>• Pour MQXCC_FAILED:                             <ol style="list-style-type: none"> <li>1. Nettoyer l'environnement</li> <li>2. Echec de MQCONN avec MQCC_FAILED, MQRC_API_EXIT_INIT_ERROR</li> </ol> </li> <li>• Pour MQXCC_*                             <ol style="list-style-type: none"> <li>1. Agir comme pour les valeurs de MQXCC_* et MQXR2_*<sup>1</sup></li> <li>2. Nettoyer l'environnement</li> </ol> </li> </ul>
e3	Avant que la fonction MQ_CONNX_EXIT 1 se termine par: <ul style="list-style-type: none"> <li>• MQXCC_ECHEC</li> <li>• MQXCC_*</li> </ul>	<ul style="list-style-type: none"> <li>• Pour MQXCC_FAILED:                             <ol style="list-style-type: none"> <li>1. Fonction MQ_TERM_EXIT de l'unité</li> <li>2. Nettoyer l'environnement</li> <li>3. Echec de l'appel MQCONN avec MQCC_FAILED, MQRC_API_EXIT_ERROR</li> </ol> </li> <li>• Pour MQXCC_*                             <ol style="list-style-type: none"> <li>1. Agir comme pour les valeurs de MQXCC_* et MQXR2_*<sup>1</sup></li> <li>2. Fonction MQ_TERM_EXIT de l'unité, si nécessaire</li> <li>3. Nettoyer l'environnement si nécessaire</li> </ol> </li> </ul>
e4	Avant que la fonction MQ_CONNX_EXIT 2 se termine par: <ul style="list-style-type: none"> <li>• MQXCC_ECHEC</li> <li>• MQXCC_*</li> </ul>	<ul style="list-style-type: none"> <li>• Pour MQXCC_FAILED:                             <ol style="list-style-type: none"> <li>1. Unité après la fonction MQ_CONNX_EXIT 1</li> <li>2. Fonction MQ_TERM_EXIT de l'unité</li> <li>3. Nettoyer l'environnement</li> <li>4. Echec de l'appel MQCONN avec MQCC_FAILED, MQRC_API_EXIT_ERROR</li> </ol> </li> <li>• Pour MQXCC_*                             <ol style="list-style-type: none"> <li>1. Agir comme pour les valeurs de MQXCC_* et MQXR2_*<sup>1</sup></li> <li>2. Unité après la fonction MQ_CONNX_EXIT 1 si l'exit n'est pas supprimé</li> <li>3. Fonction MQ_TERM_EXIT de l'unité, si nécessaire</li> <li>4. Nettoyer l'environnement si nécessaire</li> </ol> </li> </ul>



Tableau 837. Erreurs de sortie d'API et actions appropriées à effectuer (suite)

Err Pt	Description	Actions
e5	Echec de l'appel MQCONN.	<ol style="list-style-type: none"> <li>1. Transmission de MQCONN CompCode et de la raison</li> <li>2. Lecteur <i>après</i> la fonction MQ_CONNX_EXIT 2 si le <i>antérieur</i> à MQ_CONNX_EXIT 2 a abouti et que l'exit n'est pas supprimé</li> <li>3. Unité <i>après la fonction</i> MQ_CONNX_EXIT 1 si la valeur <i>antérieure</i> à MQ_CONNX_EXIT 1 a abouti et que l'exit n'est pas supprimé</li> <li>4. Fonction MQ_TERM_EXIT de l'unité</li> <li>5. Nettoyer l'environnement</li> </ol>
e6	<p><i>Après</i> l'exécution de la fonction MQ_CONNX_EXIT 2 avec:</p> <ul style="list-style-type: none"> <li>• MQXCC_ECHEC</li> <li>• MQXCC_*</li> </ul>	<ul style="list-style-type: none"> <li>• Pour MQXCC_FAILED: <ol style="list-style-type: none"> <li>1. Unité <i>après la fonction</i> MQ_CONNX_EXIT 1</li> <li>2. Terminez l'appel MQCONN avec MQCC_FAILED, MQRC_API_EXIT_ERROR</li> </ol> </li> <li>• Pour MQXCC_* <ol style="list-style-type: none"> <li>1. Agir comme pour les valeurs de MQXCC_* et MQXR2_*<sup>1</sup></li> <li>2. Unité <i>après la fonction</i> MQ_CONNX_EXIT 1, si nécessaire</li> </ol> </li> </ul>
e7	<p><i>Après</i> l'exécution de la fonction MQ_CONNX_EXIT 1 avec:</p> <ul style="list-style-type: none"> <li>• MQXCC_ECHEC</li> <li>• MQXCC_*</li> </ul>	<ul style="list-style-type: none"> <li>• Pour MQXCC_FAILED, effectuez l'appel MQCONN avec MQCC_FAILED, MQRC_API_EXIT_ERROR</li> <li>• Pour MQXCC_*, agir comme pour les valeurs de MQXCC_* et MQXR2_*<sup>1</sup></li> </ul>
e8	<p><i>Avant</i>, la fonction MQ_OPEN_EXIT 1 se termine par:</p> <ul style="list-style-type: none"> <li>• MQXCC_ECHEC</li> <li>• MQXCC_*</li> </ul>	<ul style="list-style-type: none"> <li>• Pour MQXCC_FAILED, effectuez l'appel MQOPEN avec MQCC_FAILED, MQRC_API_EXIT_ERROR</li> <li>• Pour MQXCC_*, agir comme pour les valeurs de MQXCC_* et MQXR2_*<sup>1</sup></li> </ul>
e9	<p><i>Avant</i>, la fonction MQ_OPEN_EXIT 2 se termine par:</p> <ul style="list-style-type: none"> <li>• MQXCC_ECHEC</li> <li>• MQXCC_*</li> </ul>	<ul style="list-style-type: none"> <li>• Pour MQXCC_FAILED: <ol style="list-style-type: none"> <li>1. Unité <i>après la fonction</i> MQ_OPEN_EXIT 1</li> <li>2. Terminer l'appel MQOPEN avec MQCC_FAILED, MQRC_API_EXIT_ERROR</li> </ol> </li> <li>• Pour MQXCC_*, agir comme pour les valeurs de MQXCC_* et MQXR2_*<sup>1</sup></li> </ul>
e10	Echec de l'appel MQOPEN	<ol style="list-style-type: none"> <li>1. Transmission de MQOPEN CompCode et raison</li> <li>2. Unité <i>après</i> la fonction MQ_OPEN_EXIT 2 si l'exit n'est pas supprimé</li> <li>3. Unité <i>après la fonction</i> MQ_OPEN_EXIT 1 si l'exit n'est pas supprimé et si les exits chaînés ne sont pas supprimés</li> </ol>

Tableau 837. Erreurs de sortie d'API et actions appropriées à effectuer (suite)

Err Pt	Description	Actions
e1 1	Après l'exécution de la fonction MQ_OPEN_EXIT 2 avec: <ul style="list-style-type: none"> <li>• MQXCC_ECHEC</li> <li>• MQXCC_*</li> </ul>	<ul style="list-style-type: none"> <li>• Pour MQXCC_FAILED: <ol style="list-style-type: none"> <li>1. Unité <i>après la fonction</i> MQ_OPEN_EXIT 1</li> <li>2. Terminer l'appel MQOPEN avec MQCC_FAILED, MQRC_API_EXIT_ERROR</li> </ol> </li> <li>• Pour MQXCC_* <ol style="list-style-type: none"> <li>1. Agir comme pour les valeurs de MQXCC_* et MQXR2_*<sup>1</sup></li> <li>2. Unité <i>après la fonction</i> MQ_OPEN_EXIT 1 si l'exit n'est pas supprimé</li> </ol> </li> </ul>
e2 5	Après l'exécution de la fonction MQ_DISC_EXIT 2 avec: <ul style="list-style-type: none"> <li>• MQXCC_ECHEC</li> <li>• MQXCC_*</li> </ul>	<ul style="list-style-type: none"> <li>• Pour MQXCC_FAILED: <ol style="list-style-type: none"> <li>1. Unité <i>après la fonction</i> MQ_DISC_EXIT 1</li> <li>2. Fonction MQ_TERM_EXIT de l'unité</li> <li>3. Nettoyer l'environnement d'exécution de l'exit</li> <li>4. Terminer l'appel MQDISC avec MQCC_FAILED, MQRC_API_EXIT_ERROR</li> </ol> </li> <li>• Pour MQXCC_* <ol style="list-style-type: none"> <li>1. Agir comme pour les valeurs de MQXCC_* et MQXR2_*<sup>1</sup></li> <li>2. Fonction MQ_TERM_EXIT de l'unité</li> <li>3. Nettoyer l'environnement d'exécution de l'exit</li> </ol> </li> </ul>

**Remarque :**

1. Les valeurs de MQXCC\_\* et MQXR2\_\* et les actions correspondantes sont définies dans Fonctionnement des exits des gestionnaires de files d'attente.

**Les champs ExitResponse ne sont pas définis correctement**

Cette rubrique fournit des informations sur ce qui se passerait lorsque la zone ExitResponse est définie sur une valeur autre que les valeurs prises en charge.

Si la zone ExitResponse est définie sur une valeur autre que l'une des valeurs prises en charge, les actions suivantes s'appliquent:

- Pour une fonction d'exit API *avant* MQCONN ou MQDISC:
  - La valeur ExitResponse2 est ignorée.
  - Aucune autre fonction d'exit *avant* dans la chaîne d'exit (le cas échéant) n'est appelée ; l'appel d'API lui-même n'est pas émis.
  - Pour les exits *avant* qui ont été appelés avec succès, les exits *après* sont appelés dans l'ordre inverse.
  - Si elles sont enregistrées, les fonctions d'exit de fin pour les fonctions d'exit *avant* MQCONN ou MQDISC de la chaîne qui ont été appelées avec succès sont déclenchées pour le nettoyage après ces fonctions d'exit.
  - L'appel MQCONN ou MQDISC échoue avec MQRC\_API\_EXIT\_ERROR.
- Pour une fonction d'exit API *before* IBM MQ autre que MQCONN ou MQDISC:
  - La valeur ExitResponse2 est ignorée.
  - Aucune autre fonction de conversion de données *avant* ou *après* dans la chaîne d'exit (le cas échéant) n'est appelée.

- Pour les exits *avant* qui ont été appelés avec succès, les exits *après* sont appelés dans l'ordre inverse.
- L'appel d'API IBM MQ lui-même n'est pas émis.
- L'appel de l'API IBM MQ échoue avec MQRC\_API\_EXIT\_ERROR.
- Pour une fonction d'exit API MQCONN ou MQDISC *après* :
  - La valeur ExitResponse2 est ignorée.
  - Les autres fonctions d'exit qui ont été appelées avec succès avant l'appel d'API sont appelées dans l'ordre inverse.
  - Si elles sont enregistrées, les fonctions d'exit d'arrêt pour les fonctions d'exit *avant* ou *après* MQCONN ou MQDISC de la chaîne qui ont été appelées avec succès sont exécutées pour être nettoyées après l'exit.
  - Un CompCode plus grave de MQCC\_WARNING et le CompCode renvoyé par l'exit est renvoyé à l'application.
  - Une raison de MQRC\_API\_EXIT\_ERROR est renvoyée à l'application.
  - L'appel de l'API IBM MQ a été émis.
- Pour une fonction d'exit d'appel d'API *après* IBM MQ autre que MQCONN ou MQDISC:
  - La valeur ExitResponse2 est ignorée.
  - Les autres fonctions d'exit qui ont été appelées avec succès avant l'appel d'API sont appelées dans l'ordre inverse.
  - Un CompCode plus grave de MQCC\_WARNING et le CompCode renvoyé par l'exit est renvoyé à l'application.
  - Une raison de MQRC\_API\_EXIT\_ERROR est renvoyée à l'application.
  - L'appel de l'API IBM MQ a été émis.
- Pour la fonction *before* data conversion on get exit:
  - La valeur ExitResponse2 est ignorée.
  - Les autres fonctions d'exit qui ont été appelées avec succès avant l'appel d'API sont appelées dans l'ordre inverse.
  - Le message n'est pas converti et le message non converti est renvoyé à l'application.
  - Un CompCode plus grave de MQCC\_WARNING et le CompCode renvoyé par l'exit est renvoyé à l'application.
  - Une raison de MQRC\_API\_EXIT\_ERROR est renvoyée à l'application.
  - L'appel de l'API IBM MQ a été émis.

**Remarque :** Comme l'erreur est liée à l'exit, il est préférable de renvoyer MQRC\_API\_EXIT\_ERROR que de renvoyer MQRC\_NOT\_CONVERTIS.

Si une fonction d'exit définit la zone ExitResponse2 sur une valeur autre que l'une des valeurs prises en charge, la valeur MQXR2\_DEFAULT\_CONTINUATION est utilisée à la place.

## Informations de référence de l'interface des services installables


Cette collection de rubriques fournit des informations de référence pour les services installables.

Les fonctions et les types de données sont répertoriés par ordre alphabétique au sein du groupe pour chaque type de service.

### Concepts associés

 [Services et composants installables pour UNIX, Linux et Windows](#)

 [Services et composants installables pour IBM i](#)

 [Informations de référence sur l'interface des services installables pour IBM i](#)

## Tâches associées

Extension des fonctions du gestionnaire de files d'attente

 Configuration des services installables

## Affichage des fonctions

La façon dont les fonctions des services installables sont documentées.

Pour chaque fonction, il existe une description, y compris l'identificateur de fonction (pour MQZEP).

Les *paramètres* sont répertoriés dans l'ordre dans lequel ils doivent apparaître. Ils doivent tous être présents.

Chaque nom de paramètre est suivi de son type de données. Il s'agit des types de données élémentaires décrits dans le «Types de données élémentaires», à la page 238.

L'appel de langage C est également fourni, après la description des paramètres.

## MQZ\_AUTHENTICATE\_USER-Authentification de l'utilisateur

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_5 et est appelée par le gestionnaire de files d'attente pour authentifier un utilisateur ou pour définir des zones de contexte d'identité. Il est appelé lorsque le contexte de l'application utilisateur IBM MQ est établi.

Le contexte d'application est établi lors des appels de connexion au point où le contexte utilisateur de l'application est initialisé et à chaque point où le contexte utilisateur de l'application est modifié. Chaque fois qu'un appel de connexion est effectué, les informations de contexte utilisateur de l'application sont réacquises dans la zone *IdentityContext*.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_AUTHENTICATE\_USER.

## Syntaxe

MQZ\_AUTHENTICATE\_USER ( *QMgrName* , *SecurityParms* , *ApplicationContext* , *IdentityContext* , *CorrelationPtr* , *ComponentData* , *Continuation* , *CompCode* , *Raison* )

## Paramètres

### QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

### SecurityParms

Type: MQCSP-entrée

Paramètres de sécurité. Données relatives à l'ID utilisateur, au mot de passe et au type d'authentification. Si l'attribut *AuthenticationType* de la structure MQCSP est spécifié en tant que MQCSP\_AUTH\_USER\_ID\_AND\_PWD, l'ID utilisateur et le mot de passe sont comparés aux zones équivalentes du paramètre *IdentityContext* (MQZIC) pour déterminer s'ils correspondent. Pour plus d'informations, voir «MQCSP-Paramètres de sécurité», à la page 346.

Lors d'un appel MQCONN MQI, ce paramètre contient des valeurs null ou des valeurs par défaut.

### ApplicationContext

Type: MQZAC-entrée

Contexte d'application. Données relatives à l'application appelante. Voir MQZAC-Contexte d'application pour plus de détails.

Lors de chaque appel MQCONN ou MQCONNX MQI, les informations de contexte utilisateur de la structure MQZAC sont réacquises.

### **IdentityContext**

Type: MQZIC-entrée / sortie

Contexte d'identité. Lors de l'entrée dans la fonction d'authentification de l'utilisateur, identifie le contexte d'identité en cours. La fonction d'authentification de l'utilisateur peut changer cela, auquel cas le gestionnaire de files d'attente adopte le nouveau contexte d'identité. Voir [Contexte MQZIC-Identity](#) pour plus de détails sur la structure MQZIC.

### **CorrelationPtr**

Type: MQPTR-sortie

Pointeur de corrélation. Indique l'adresse des données de corrélation. Ce pointeur est ensuite transmis à d'autres appels OAM.

### **ComponentData**

Type: MQBYTE x ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors du prochain appel de l'une des fonctions de ce composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre de longueur ComponentData de l'appel MQZ\_INIT\_AUTHORITY.

### **Continuation**

Type : MQLONG - sortie

Indicateur de continuation. Vous pouvez spécifier les valeurs suivantes:

#### **MQZCI\_DEFAULT**

Continuation dépendante des autres composants.

#### **MQZCI\_ARRET**

Ne passez pas au composant suivant.

### **CompCode**

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

### **Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

Pour plus d'informations sur ces codes anomalie, voir [Messages et codes anomalie](#).

## Appel C

```
MQZ_AUTHENTICATE_USER (QMgrName, SecurityParms, ApplicationContext,  
                        IdentityContext, &CorrelationPtr, ComponentData,  
                        &Continuation, &CompCode, &Reason);
```

Déclarez les paramètres transmis au service comme suit:

```
MQCHAR48  QMgrName;           /* Queue manager name */  
MQCSP     SecurityParms;     /* Security parameters */  
MQZAC     ApplicationContext; /* Application context */  
MQZIC     IdentityContext;   /* Identity context */  
MQPTR     CorrelationPtr;    /* Correlation pointer */  
MQBYTE    ComponentData[n];  /* Component data */  
MQLONG    Continuation;     /* Continuation indicator set by  
                             component */  
MQLONG    CompCode;         /* Completion code */  
MQLONG    Reason;          /* Reason code qualifying CompCode */
```

## MQZ\_CHECK\_AUTHORITY-Vérification des droits

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_1 et est démarrée par le gestionnaire de files d'attente pour vérifier si une entité est autorisée à effectuer une ou plusieurs actions particulières sur un objet spécifié.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_CHECK\_AUTHORITY.

### Syntaxe

```
MQZ_CHECK_AUTHORITY( QMgrName , EntityName , EntityType , ObjectName ,  
ObjectType , Authority , ComponentData , Continuation , CompCode , Reason )
```

### Paramètres

#### QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

#### EntityName

Type: MQCHAR12 -entrée

Nom de l'entité. Nom de l'entité dont l'autorisation sur l'objet doit être vérifiée. La longueur maximale de la chaîne est de 12 caractères ; si elle est plus courte, elle est remplie à droite avec des blancs. Le nom ne se termine pas par un caractère NULL.

Il n'est pas essentiel que cette entité soit connue du service de sécurité sous-jacent. S'il n'est pas connu, les autorisations du groupe spécial **nobody** (auquel toutes les entités sont supposées appartenir) sont utilisées pour la vérification. Un nom vide est valide et peut être utilisé de cette manière.

#### EntityType

Type : MQLONG - entrée

Type d'entité. Type d'entité spécifié par EntityName. Il doit s'agir de l'une des valeurs suivantes:

**MQZAET\_PRINCIPAL**

Entité.

**GRUPE MQZAET\_GROUP**

**ObjectName**

Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet auquel l'accès est requis. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT\_Q\_MGR, ce nom est identique à *QMgrName*.

**ObjectType**

Type : MQLONG - entrée

Type d'objet. Type d'entité spécifié par *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

**INFO MQOT\_AUTH\_INFO**

Informations d'authentification.

**CANAL\_MQTON**

Canal.

**MQOT\_CLNTCONN\_CHANNEL**

Canal de connexion client.

**MQOT\_PROGRAMME d'écoute**

Programme d'écoute.

**MQOT\_NAMELIST**

NAMELIST.

**PROCESSUS MQ**

Définition de processus.

**MQOT\_Q**

File d'attente.

**MQOT\_Q\_DIR**

Gestionnaire de files d'attente.

**SERVICE MQOT**

Service.

**Droits d'accès**

Type : MQLONG - entrée

Droits à vérifier. Si une autorisation est vérifiée, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO\_\*). Si plusieurs autorisations sont vérifiées, il s'agit du OU bit à bit des constantes MQZAO\_\* correspondantes.

Les autorisations suivantes s'appliquent à l'utilisation des appels MQI:

**MQZAO\_CONNECT**

Possibilité d'utiliser l'appel MQCONN.

**MQZAO\_PARCOURIR**

Possibilité d'utiliser l'appel MQGET avec une option de navigation.

Cela permet de spécifier l'option MQGMO\_BROWSE\_FIRST, MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR ou MQGMO\_BROWSE\_NEXT sur l'appel MQGET.

**MQZAO\_ENTREE**

Entité. Possibilité d'utiliser l'appel MQGET avec une option d'entrée.

Cela permet à l'option MQOO\_INPUT\_SHARED, MQOO\_INPUT\_EXCLUSIVE ou MQOO\_INPUT\_AS\_Q\_DEF d'être spécifiée sur l'appel MQOPEN.

**MQZAO\_OUTPUT**

Possibilité d'utiliser l'appel MQPUT.

Cela permet à l'option MQOO\_OUTPUT d'être spécifiée sur l'appel MQOPEN.

**MQZAO\_INQUIRE**

Possibilité d'utiliser l'appel MQINQ.

Cela permet de spécifier l'option MQOO\_INQUIRE sur l'appel MQOPEN.

**MQZAO\_SET**

Possibilité d'utiliser l'appel MQSET.

Cela permet de spécifier l'option MQOO\_SET sur l'appel MQOPEN.

**MQZAO\_PASS\_IDENTITY\_CONTEXT**

Possibilité de transmettre le contexte d'identité.

Cela permet de spécifier l'option MQOO\_PASS\_IDENTITY\_CONTEXT sur l'appel MQOPEN et l'option MQPMO\_PASS\_IDENTITY\_CONTEXT sur les appels MQPUT et MQPUT1 .

**MQZAO\_PASS\_ALL\_CONTEXT**

Possibilité de transmettre tout le contexte.

Cela permet de spécifier l'option MQOO\_PASS\_ALL\_CONTEXT sur l'appel MQOPEN et l'option MQPMO\_PASS\_ALL\_CONTEXT sur les appels MQPUT et MQPUT1 .

**MQZAO\_SET\_IDENTITY\_CONTEXT**

Possibilité de définir le contexte d'identité.

Cela permet de spécifier l'option MQOO\_SET\_IDENTITY\_CONTEXT sur l'appel MQOPEN et l'option MQPMO\_SET\_IDENTITY\_CONTEXT sur les appels MQPUT et MQPUT1 .

**MQZAO\_SET\_ALL\_CONTEXT**

Possibilité de définir tous les contextes.

Cela permet de spécifier l'option MQOO\_SET\_ALL\_CONTEXT sur l'appel MQOPEN et l'option MQPMO\_SET\_ALL\_CONTEXT sur les appels MQPUT et MQPUT1 .

**MQZAO\_ALTERNATE\_USER\_AUTHORITY**

Possibilité d'utiliser des droits utilisateur de remplacement.

Cela permet de spécifier l'option MQOO\_ALTERNATE\_USER\_AUTHORITY sur l'appel MQOPEN et l'option MQPMO\_ALTERNATE\_USER\_AUTHORITY sur l'appel MQPUT1 .

**MQZAO\_ALL\_MQI**

Toutes les autorisations MQI.

Cela active toutes les autorisations.

Les autorisations suivantes s'appliquent à l'administration d'un gestionnaire de files d'attente:

**MQZAO\_CREER**

Possibilité de créer des objets d'un type spécifié.

**MQZAO\_DELETE**

Possibilité de supprimer un objet spécifié.

**MQZAO\_DISPLAY**

Possibilité d'afficher les attributs d'un objet spécifié.

**MODIFICATION MQZAO\_DE**

Possibilité de modifier les attributs d'un objet spécifié.

**MQZAO\_CLEAR**

Possibilité de supprimer tous les messages d'une file d'attente spécifiée.

**MQZAO\_AUTORISATION**

Possibilité d'autoriser d'autres utilisateurs pour un objet spécifié.

**CONTROLE MQZ**

Possibilité de démarrer ou d'arrêter un programme d'écoute, un service ou un objet de canal non client, et possibilité de lancer une commande ping sur un objet de canal non client.



### **MQZAO\_CONTRÔLE\_ÉTENDU**

Possibilité de réinitialiser un numéro de séquence ou de résoudre un message en attente de validation sur un objet canal non client.

### **MQZAO\_ALL\_ADMIN**

Possibilité de définir le contexte d'identité.

Toutes les autorisations d'administration, autres que MQZAO\_CREATE.

Les autorisations suivantes s'appliquent à l'utilisation de l'interface MQI et à l'administration d'un gestionnaire de files d'attente:

### **MQZAO\_ALL**

Toutes les autorisations, autres que MQZAO\_CREATE.

### **MQZAO\_AUCUN**

Aucune autorisation.

### **ComponentData**

Type: MQBYTE x ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

### **Continuation**

Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

### **MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_CHECK\_AUTHORITY, cela a le même effet que MQZCI\_STOP.

### **MQZCI\_CONTINUER**

Passez au composant suivant.

### **MQZCI\_ARRET**

Ne passez pas au composant suivant.

Si l'appel à un composant échoue (c'est-à-dire que *CompCode* renvoie MQCC\_FAILED) et que le paramètre *Continuation* est MQZCI\_DEFAULT ou MQZCI\_CONTINUE, le gestionnaire de files d'attente continue d'appeler d'autres composants s'il en existe.

Si l'appel aboutit (c'est-à-dire si *CompCode* renvoie MQCC\_OK), aucun autre composant n'est appelé, quelle que soit la valeur de *Continuation*.

Si l'appel échoue et que le paramètre *Continuation* est MQZCI\_STOP, aucun autre composant n'est appelé et l'erreur est renvoyée au gestionnaire de files d'attente. Les composants ne connaissant pas les appels précédents, le paramètre *Continuation* est toujours défini sur MQZCI\_DEFAULT avant l'appel.

### **CompCode**

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

### **MQCC\_OK**

Achèvement réussi.

### **MQCC\_FAILED**

Echec de l'appel.

## Motif

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

### **MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') Non autorisé pour l'accès.

### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

## Appel C

```
MQZ_CHECK_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,  
                    ObjectType, Authority, ComponentData,  
                    &Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48  QMgrName;           /* Queue manager name */  
MQCHAR12  EntityName;        /* Entity name */  
MQLONG    EntityType;        /* Entity type */  
MQCHAR48  ObjectName;        /* Object name */  
MQLONG    ObjectType;        /* Object type */  
MQLONG    Authority;         /* Authority to be checked */  
MQBYTE    ComponentData[n]; /* Component data */  
MQLONG    Continuation;      /* Continuation indicator set by  
                             component */  
MQLONG    CompCode;          /* Completion code */  
MQLONG    Reason;           /* Reason code qualifying CompCode */
```

## **MQZ\_CHECK\_AUTHORITY\_2 -Vérification des droits (étendu)**

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_2 et est démarrée par le gestionnaire de files d'attente pour vérifier si une entité est autorisée à effectuer une ou plusieurs actions sur un objet spécifié.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_CHECK\_AUTHORITY.

MQZ\_CHECK\_AUTHORITY\_2 est similaire à MQZ\_CHECK\_AUTHORITY, mais avec le paramètre **EntityName** remplacé par le paramètre **EntityData**.

### **Syntaxe**

```
MQZ_CHECK_AUTHORITY_2( QMgrName , EntityData , EntityType , ObjectName ,  
ObjectType , Authority , ComponentData , Continuation , CompCode , Reason )
```

### **Paramètres**

#### **QMgrName**

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

### **EntityData**

Type: MQZED-entrée

Données d'entité. Données relatives à l'entité avec autorisation sur l'objet à vérifier. Pour plus d'informations, voir «MQZED-Descripteur d'entité», à la page 1759.

Il n'est pas essentiel que cette entité soit connue du service de sécurité sous-jacent. S'il n'est pas connu, les autorisations du groupe spécial **nobody** (auquel toutes les entités sont supposées appartenir) sont utilisées pour la vérification. Un nom vide est valide et peut être utilisé de cette manière.

### **EntityType**

Type : MQLONG - entrée

Type d'entité. Type d'entité spécifié par *EntityData*. Il doit s'agir de l'une des valeurs suivantes:

#### **MQZAET\_PRINCIPAL**

Entité.

#### **GROUPE MQZAET\_GROUP**

### **ObjectName**

Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet auquel l'accès est requis. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT\_Q\_MGR, ce nom est identique à *QMgrName*.

### **ObjectType**

Type : MQLONG - entrée

Type d'objet. Type d'entité spécifié par *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

#### **INFO MQOT\_AUTH\_INFO**

Informations d'authentification.

#### **CANAL\_MQTON**

Canal.

#### **MQOT\_CLNTCONN\_CHANNEL**

Canal de connexion client.

#### **MQOT\_PROGRAMME d'écoute**

Programme d'écoute.

#### **MQOT\_NAMELIST**

NAMELIST.

#### **PROCESSUS MQ**

Définition de processus.

#### **MQOT\_Q**

File d'attente.

#### **MQOT\_Q\_DIR**

Gestionnaire de files d'attente.

#### **SERVICE MQOT**

Service.

**MQOT\_TOPIC**

:NONE.

**Droits d'accès**

Type : MQLONG - entrée

Droits à vérifier. Si une autorisation est vérifiée, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO\_\*). Si plusieurs autorisations sont vérifiées, il s'agit du OU bit à bit des constantes MQZAO\_\* correspondantes.

Les autorisations suivantes s'appliquent à l'utilisation des appels MQI:

**MQZAO\_CONNECT**

Possibilité d'utiliser l'appel MQCONN.

**MQZAO\_PARCOURIR**

Possibilité d'utiliser l'appel MQGET avec une option de navigation.

Cela permet de spécifier l'option MQGMO\_BROWSE\_FIRST, MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR ou MQGMO\_BROWSE\_NEXT sur l'appel MQGET.

**MQZAO\_ENTREE**

Entité. Possibilité d'utiliser l'appel MQGET avec une option d'entrée.

Cela permet à l'option MQOO\_INPUT\_SHARED, MQOO\_INPUT\_EXCLUSIVE ou MQOO\_INPUT\_AS\_Q\_DEF d'être spécifiée sur l'appel MQOPEN.

**MQZAO\_OUTPUT**

Possibilité d'utiliser l'appel MQPUT.

Cela permet à l'option MQOO\_OUTPUT d'être spécifiée sur l'appel MQOPEN.

**MQZAO\_INQUIRE**

Possibilité d'utiliser l'appel MQINQ.

Cela permet de spécifier l'option MQOO\_INQUIRE sur l'appel MQOPEN.

**MQZAO\_SET**

Possibilité d'utiliser l'appel MQSET.

Cela permet de spécifier l'option MQOO\_SET sur l'appel MQOPEN.

**MQZAO\_PASS\_IDENTITY\_CONTEXT**

Possibilité de transmettre le contexte d'identité.

Cela permet de spécifier l'option MQOO\_PASS\_IDENTITY\_CONTEXT sur l'appel MQOPEN et l'option MQPMO\_PASS\_IDENTITY\_CONTEXT sur les appels MQPUT et MQPUT1 .

**MQZAO\_PASS\_ALL\_CONTEXT**

Possibilité de transmettre tout le contexte.

Cela permet de spécifier l'option MQOO\_PASS\_ALL\_CONTEXT sur l'appel MQOPEN et l'option MQPMO\_PASS\_ALL\_CONTEXT sur les appels MQPUT et MQPUT1 .

**MQZAO\_SET\_IDENTITY\_CONTEXT**

Possibilité de définir le contexte d'identité.

Cela permet de spécifier l'option MQOO\_SET\_IDENTITY\_CONTEXT sur l'appel MQOPEN et l'option MQPMO\_SET\_IDENTITY\_CONTEXT sur les appels MQPUT et MQPUT1 .

**MQZAO\_SET\_ALL\_CONTEXT**

Possibilité de définir tous les contextes.

Cela permet de spécifier l'option MQOO\_SET\_ALL\_CONTEXT sur l'appel MQOPEN et l'option MQPMO\_SET\_ALL\_CONTEXT sur les appels MQPUT et MQPUT1 .

**MQZAO\_ALTERNATE\_USER\_AUTHORITY**

Possibilité d'utiliser des droits utilisateur de remplacement.

Cela permet de spécifier l'option MQOO\_ALTERNATE\_USER\_AUTHORITY sur l'appel MQOPEN et l'option MQPMO\_ALTERNATE\_USER\_AUTHORITY sur l'appel MQPUT1 .

#### **MQZAO\_ALL\_MQI**

Toutes les autorisations MQI.

Cela active toutes les autorisations.

Les autorisations suivantes s'appliquent à l'administration d'un gestionnaire de files d'attente:

#### **MQZAO\_CREER**

Possibilité de créer des objets d'un type spécifié.

#### **MQZAO\_DELETE**

Possibilité de supprimer un objet spécifié.

#### **MQZAO\_DISPLAY**

Possibilité d'afficher les attributs d'un objet spécifié.

#### **MODIFICATION MQZAO\_DE**

Possibilité de modifier les attributs d'un objet spécifié.

#### **MQZAO\_CLEAR**

Possibilité de supprimer tous les messages d'une file d'attente spécifiée.

#### **MQZAO\_AUTORISATION**

Possibilité d'autoriser d'autres utilisateurs pour un objet spécifié.

#### **CONTROLE MQZ**

Possibilité de démarrer ou d'arrêter un programme d'écoute, un service ou un objet de canal non client, et possibilité de lancer une commande ping sur un objet de canal non client.

#### **MQZAO\_CONTRÔLE\_ÉTENDU**

Possibilité de réinitialiser un numéro de séquence ou de résoudre un message en attente de validation sur un objet canal non client.

#### **MQZAO\_ALL\_ADMIN**

Possibilité de définir le contexte d'identité.

Toutes les autorisations d'administration, autres que MQZAO\_CREATE.

Les autorisations suivantes s'appliquent à l'utilisation de l'interface MQI et à l'administration d'un gestionnaire de files d'attente:

#### **MQZAO\_ALL**

Toutes les autorisations, autres que MQZAO\_CREATE.

#### **MQZAO\_AUCUN**

Aucune autorisation.

### **ComponentData**

Type: MQBYTE x ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

### **Continuation**

Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

#### **MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_CHECK\_AUTHORITY, cela a le même effet que MQZCI\_STOP.

## **MQZCI\_CONTINUER**

Passez au composant suivant.

## **MQZCI\_ARRET**

Ne passez pas au composant suivant.

### **CompCode**

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

### **Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') Non autorisé pour l'accès.

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

#### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

## **Appel C**

```
MQZ_CHECK_AUTHORITY_2 (QMgrName, &EntityData, EntityType,  
                        ObjectName, ObjectType, Authority, ComponentData,  
                        &Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48 QMgrName;          /* Queue manager name */  
MQZED EntityData;          /* Entity data */  
MQLONG EntityType;         /* Entity type */  
MQCHAR48 ObjectName;       /* Object name */  
MQLONG ObjectType;         /* Object type */  
MQLONG Authority;          /* Authority to be checked */  
MQBYTE ComponentData[n];   /* Component data */  
MQLONG Continuation;       /* Continuation indicator set by  
                             component */  
MQLONG CompCode;           /* Completion code */  
MQLONG Reason;             /* Reason code qualifying CompCode */
```

## **MQZ\_CHECK\_PRIVILEGED-vérifie si l'utilisateur est privilégié**

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_6 et est appelée par le gestionnaire de files d'attente pour déterminer si un utilisateur spécifié est un utilisateur privilégié.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_CHECK\_PRIVILEGED.

## Syntaxe

`MQZ_CHECK_PRIVILEGED( QMgrName , EntityData , EntityType , ComponentData , Continuation , CompCode , Reason )`

## Paramètres

### QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

### EntityData

Type: MQZED-entrée

Données d'entité. Données relatives à l'entité à vérifier. Pour plus d'informations, voir «MQZED- Descripteur d'entité», à la page 1759.

### EntityType

Type : MQLONG - entrée

Type d'entité. Type d'entité spécifié par EntityData. Il doit s'agir de l'une des valeurs suivantes:

#### **MQZAET\_PRINCIPAL**

Entité.

#### **GROUPE MQZAET\_GROUP**

### ComponentData

Type: MQBYTEExComponentDataLength -entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

### Continuation

Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

#### **MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_CHECK\_AUTHORITY, cela a le même effet que MQZCI\_STOP.

#### **MQZCI\_CONTINUER**

Passez au composant suivant.

#### **MQZCI\_ARRET**

Ne passez pas au composant suivant.

Si l'appel à un composant échoue (c'est-à-dire que *CompCode* renvoie MQCC\_FAILED) et que le paramètre *Continuation* est MQZCI\_DEFAULT ou MQZCI\_CONTINUE, le gestionnaire de files d'attente continue d'appeler d'autres composants s'il en existe.

Si l'appel aboutit (c'est-à-dire si *CompCode* renvoie MQCC\_OK), aucun autre composant n'est appelé, quelle que soit la valeur de *Continuation* .

Si l'appel échoue et que le paramètre *Continuation* est MQZCI\_STOP, aucun autre composant n'est appelé et l'erreur est renvoyée au gestionnaire de files d'attente. Les composants ne connaissant pas les appels précédents, le paramètre *Continuation* est toujours défini sur MQZCI\_DEFAULT avant l'appel.

### CompCode

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

#### MQCC\_OK

Achèvement réussi.

#### MQCC\_FAILED

Echec de l'appel.

### Motif

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### MQRC\_NONE

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### MQRC\_NOT\_PRIVILEGED

(2584, X'A18') Cet utilisateur n'est pas un ID utilisateur privilégié.

#### MQRC\_UNKNOWN\_ENTITY

(2292, X'8F4') Entité inconnue du service.

#### MQRC\_SERVICE\_ERROR

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

#### MQRC\_SERVICE\_NOT\_AVAILABLE

(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

## Appel C

```
MQZ_CHECK_PRIVILEGED (QMgrName, &EntityData, EntityType,  
                      ComponentData, &Continuation,  
                      &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48  QMgrName;          /* Queue manager name */  
MQZED     EntityData;       /* Entity name */  
MQLONG    EntityType;       /* Entity type */  
MQBYTE    ComponentData[n]; /* Component data */  
MQLONG    Continuation;     /* Continuation indicator set by  
                           component */  
MQLONG    CompCode;         /* Completion code */  
MQLONG    Reason;           /* Reason code qualifying CompCode */
```

## MQZ\_COPY\_ALL\_AUTHORITY-Copie de tous les droits

Cette fonction est fournie par un composant de service d'autorisation. Il est démarré par le gestionnaire de files d'attente pour copier toutes les autorisations actuellement en vigueur pour un objet de référence dans un autre objet.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_COPY\_ALL\_AUTHORITY.



## Syntaxe

MQZ\_COPY\_ALL\_AUTHORITY( *QMgrName* , *RefObjectName* , *ObjectName* , *ObjectType* , *ComponentData* , *Continuation* , *CompCode* , *Reason* )

## Paramètres

### QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

### Nom RefObject

Type: MQCHAR48 -entrée

Nom de l'objet de référence. Nom de l'objet de référence dont les autorisations doivent être copiées. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

### ObjectName

Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet pour lequel les accès doivent être définis. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

### ObjectType

Type : MQLONG - entrée

Type d'objet. Type d'entité spécifié par *RefObjectName* et *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

#### **INFO MQOT\_AUTH\_INFO**

Informations d'authentification.

#### **CANAL\_MQTON**

Canal.

#### **MQOT\_CLNTCONN\_CHANNEL**

Canal de connexion client.

#### **MQOT\_PROGRAMME d'écoute**

Programme d'écoute.

#### **MQOT\_NAMELIST**

NAMELIST.

#### **PROCESSUS MQ**

Définition de processus.

#### **MQOT\_Q**

File d'attente.

#### **MQOT\_Q\_DIR**

Gestionnaire de files d'attente.

#### **SERVICE MQOT**

Service.

#### **MQOT\_TOPIC**

:NONE.

### ComponentData

Type: MQBYTEExComponentDataLength -entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre de longueur `ComponentData` de l'appel `MQZ_INIT_AUTHORITY`.

### Continuation

Type : `MQLONG` - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

#### **MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour `MQZ_CHECK_AUTHORITY`, cela a le même effet que `MQZCI_STOP`.

#### **MQZCI\_CONTINUER**

Passez au composant suivant.

#### **MQZCI\_ARRET**

Ne passez pas au composant suivant.

### CompCode

Type : `MQLONG` - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

### Motif

Type : `MQLONG` - sortie

Code anomalie qualifiant `CompCode`.

Si `CompCode` est `MQCC_OK`:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si `CompCode` est `MQCC_FAILED`:

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

#### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

#### **MQRC\_UNKNOWN\_REF\_OBJECT**

(2294, X'8F6') Objet de référence inconnu.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

## Appel C

```
MQZ_COPY_ALL_AUTHORITY (QMgrName, RefObjectName, ObjectName, ObjectType,  
                        ComponentData, &Continuation, &CompCode,  
                        &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48 QMgrName;          /* Queue manager name */  
MQCHAR48 RefObjectName;     /* Reference object name */  
MQCHAR48 ObjectName;       /* Object name */
```

```

MQLONG      ObjectType;          /* Object type */
MQBYTE      ComponentData[n];  /* Component data */
MQLONG      Continuation;      /* Continuation indicator set by
                               component */
MQLONG      CompCode;          /* Completion code */
MQLONG      Reason;           /* Reason code qualifying CompCode */

```

## MQZ\_DELETE\_AUTHORITY-Suppression des droits

Cette fonction est fournie par un composant de service d'autorisation et démarrée par le gestionnaire de files d'attente pour supprimer toutes les autorisations associées à l'objet spécifié.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_DELETE\_AUTHORITY.

### Syntaxe

```

MQZ_DELETE_AUTHORITY( QMgrName , ObjectName , ObjectType , ComponentData ,
Continuation , CompCode , Reason )

```

### Paramètres

#### QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

#### ObjectName

Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet pour lequel les accès doivent être supprimés. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT\_Q\_MGR, ce nom est identique à *QMgrName*.

#### ObjectType

Type : MQLONG - entrée

Type d'objet. Type d'entité spécifié par *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

#### **INFO MQOT\_AUTH\_INFO**

Informations d'authentification.

#### **CANAL\_MQTON**

Canal.

#### **MQOT\_CLNTCONN\_CHANNEL**

Canal de connexion client.

#### **MQOT\_PROGRAMME d'écoute**

Programme d'écoute.

#### **MQOT\_NAMELIST**

NAMELIST.

#### **PROCESSUS MQ**

Définition de processus.

#### **MQOT\_Q**

File d'attente.

#### **MQOT\_Q\_DIR**

Gestionnaire de files d'attente.

**SERVICE MQOT**

Service.

**MQOT\_TOPIC**

:NONE.

**ComponentData**

Type: MQBYTE x ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre de longueur ComponentData de l'appel MQZ\_INIT\_AUTHORITY.

**Continuation**

Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

**MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_CHECK\_AUTHORITY, cela a le même effet que MQZCI\_STOP.

**MQZCI\_CONTINUER**

Passez au composant suivant.

**MQZCI\_ARRET**

Ne passez pas au composant suivant.

**CompCode**

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

**MQCC\_OK**

Achèvement réussi.

**MQCC\_FAILED**

Echec de l'appel.

**Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

**MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

**Appel C**

```
MQZ_DELETE_AUTHORITY (QMgName, ObjectName, ObjectType, ComponentData,  
&Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48  QMgrName;          /* Queue manager name */
MQCHAR48  ObjectName;      /* Object name */
MQLONG    ObjectType;     /* Object type */
MQBYTE    ComponentData[n]; /* Component data */
MQLONG    Continuation;   /* Continuation indicator set by
                           component */
MQLONG    CompCode;       /* Completion code */
MQLONG    Reason;        /* Reason code qualifying CompCode */
```

## MQZ\_ENUMERATE\_AUTHORITY\_DATA-Énumération des données de droits d'accès

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_4 et est démarrée à plusieurs reprises par le gestionnaire de files d'attente pour extraire toutes les données de droits d'accès qui correspondent aux critères de sélection spécifiés lors du premier appel.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_ENUMERATE\_AUTHORITY\_DATA.

### Syntaxe

```
MQZ_ENUMERATE_AUTHORITY_DATA( QMgrName , StartEnumeration , Filter ,
AuthorityBufferLength , AuthorityBuffer , AuthorityDataLength , ComponentData ,
Continuation , CompCode , Reason )
```

### Paramètres

#### QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

#### StartEnumeration

Type : MQLONG - entrée

Indicateur signalant si l'appel peut démarrer l'énumération. Indique si l'appel peut démarrer l'énumération des données de droits d'accès ou continuer l'énumération des données de droits d'accès démarrée par un appel précédent à MQZ\_ENUMERATE\_AUTHORITY\_DATA. La valeur est l'une des suivantes:

##### DEMARRAGE MQZ

Démarrer l'énumération. L'appel est démarré avec cette valeur pour démarrer l'énumération des données de droits d'accès. Le paramètre **Filter** indique les critères de sélection à utiliser pour sélectionner les données de droits d'accès renvoyées par cet appel et les appels successifs.

##### MQZSE\_CONTINUE

Continuez l'énumération. L'appel est lancé avec cette valeur pour continuer l'énumération des données de droits. Le paramètre **Filter** est ignoré dans ce cas et peut être spécifié en tant que pointeur null (les critères de sélection sont déterminés par le paramètre **Filter** spécifié par l'appel qui a défini *StartEnumeration* sur MQZSE\_START).

#### Filtre

Type: MQZAD-entrée

filtrer. Si *StartEnumeration* est MQZSE\_START, *Filter* indique les critères de sélection à utiliser pour sélectionner les données de droits d'accès à renvoyer. Si *Filter* est le pointeur null, aucun critère de sélection n'est utilisé, c'est-à-dire que toutes les données de droits d'accès sont renvoyées.

Pour plus de détails sur les critères de sélection pouvant être utilisés, voir «MQZAD-Données de droits d'accès», à la page 1756 .

Si *StartEnumeration* est MQZSE\_CONTINUE, *Filter* est ignoré et peut être spécifié en tant que pointeur null.

### **AuthorityBufferLongueur**

Type : MQLONG - entrée

Longueur de *AuthorityBuffer*. Il s'agit de la longueur en octets du paramètre **AuthorityBuffer** . La mémoire tampon des droits d'accès doit être suffisamment grande pour contenir les données à renvoyer.

### **AuthorityBuffer**

Type: MQZAD-sortie

Données de droits d'accès. Il s'agit de la mémoire tampon dans laquelle les données de droits d'accès sont renvoyées. La mémoire tampon doit être suffisamment grande pour accueillir une structure MQZAD, une structure MQZED, plus le nom d'entité le plus long et le nom de domaine le plus long définis.

**Remarque :** Remarque: Ce paramètre est défini en tant que MQZAD, car MQZAD se produit toujours au début de la mémoire tampon. Toutefois, si la mémoire tampon est déclarée en tant que MQZAD, elle sera trop petite-elle doit être plus grande qu'une MQZAD pour pouvoir accueillir les noms d'entité et de domaine MQZAD, MQZED.

### **AuthorityDataLongueur**

Type : MQLONG - sortie

Longueur des données renvoyées dans *AuthorityBuffer*. Si la mémoire tampon des droits est trop petite, *AuthorityDataLength* est défini sur la longueur de la mémoire tampon requise et l'appel renvoie le code achèvement MQCC\_FAILED et le code anomalie MQRC\_BUFFER\_LENGTH\_ERROR.

### **ComponentData**

Type: MQBYTE x ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre de longueur ComponentData de l'appel MQZ\_INIT\_AUTHORITY.

### **Continuation**

Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

#### **MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_ENUMERATE\_AUTHORITY\_DATA, cela a le même effet que MQZCI\_CONTINUE.

#### **MQZCI\_CONTINUER**

Passez au composant suivant.

#### **MQZCI\_ARRET**

Ne passez pas au composant suivant.

### **CompCode**

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

#### **MQCC\_OK**

Achèvement réussi.

## **MQCC\_FAILED**

Echec de l'appel.

### **Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

## **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

## **MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

## **MQRC\_NO\_DATA\_XX\_ENCODE\_CASE\_ONE disponible**

(2379, X'94B') Aucune donnée disponible.

## **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

## **Appel C**

```
MQZ_ENUMERATE_AUTHORITY_DATA (QMgrName, StartEnumeration, &Filter,  
                               AuthorityBufferLength,  
                               &AuthorityBuffer,  
                               &AuthorityDataLength, ComponentData,  
                               &Continuation, &CompCode,  
                               &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48  QMgrName;           /* Queue manager name */  
MQLONG    StartEnumeration;   /* Flag indicating whether call should  
                               start enumeration */  
MQZAD     Filter;             /* Filter */  
MQLONG    AuthorityBufferLength; /* Length of AuthorityBuffer */  
MQZAD     AuthorityBuffer;    /* Authority data */  
MQLONG    AuthorityDataLength; /* Length of data returned in  
                               AuthorityBuffer */  
MQBYTE    ComponentData[n];   /* Component data */  
MQLONG    Continuation;       /* Continuation indicator set by  
                               component */  
MQLONG    CompCode;           /* Completion code */  
MQLONG    Reason;             /* Reason code qualifying CompCode */
```

## **MQZ\_FREE\_USER-Utilisateur libre**

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_5 et est démarrée par le gestionnaire de files d'attente pour libérer la ressource allouée associée.

Elle est démarrée lorsqu'une application a fini de s'exécuter dans tous les contextes utilisateur, par exemple lors d'un appel MQDISC MQI.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_FREE\_USER.

### **Syntaxe**

```
MQZ_FREE_USER( QMgrName , FreeParms , ComponentData , Continuation , CompCode ,  
Reason )
```

## Paramètres

### QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

### FreeParms

Type: MQZFP-entrée

Paramètres libres. Structure contenant des données relatives à la ressource à libérer. Pour plus d'informations, voir «MQZFP-Paramètres libres», à la page 1761.

### ComponentData

Type: MQBYTE x ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre de longueur ComponentData de l'appel MQZ\_INIT\_AUTHORITY.

### Continuation

Type : MQLONG - sortie

Indicateur de continuation. Les valeurs suivantes peuvent être spécifiées :

#### **MQZCI\_DEFAULT**

Continuation dépendante des autres composants.

#### **MQZCI\_ARRET**

Ne passez pas au composant suivant.

### CompCode

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

### Motif

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).



## Appel C

```
MQZ_AUTHENTICATE_USER (QMgrName, SecurityParms, ApplicationContext,  
IdentityContext, CorrelationPtr, ComponentData,  
&Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48  QMgrName;           /* Queue manager name */  
MQZFP     FreeParms;        /* Resource to be freed */  
MQBYTE    ComponentData[n]; /* Component data */  
MQLONG    Continuation;     /* Continuation indicator set by  
                           component */  
MQLONG    CompCode;        /* Completion code */  
MQLONG    Reason;         /* Reason code qualifying CompCode */
```

## MQZ\_GET\_AUTHORITY-Obtention des droits

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_1 et est démarrée par le gestionnaire de files d'attente pour extraire les droits dont une entité dispose pour accéder à l'objet spécifié, y compris (si l'entité est un principal) les droits détenus par les groupes dont le principal est membre. Les droits des profils génériques sont inclus dans le jeu de droits renvoyé.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_GET\_AUTHORITY.

### Syntaxe

```
MQZ_GET_AUTHORITY( QMgrName , EntityName , EntityType , ObjectName ,  
ObjectType , Authority , ComponentData , Continuation , CompCode , Reason )
```

### Paramètres

#### QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

#### EntityName

Type: MQCHAR12 -entrée

Nom de l'entité. Nom de l'entité dont l'accès à l'objet doit être extrait. La longueur maximale de la chaîne est de 12 caractères ; si elle est plus courte, elle est remplie à droite avec des blancs. Le nom ne se termine pas par un caractère NULL.

#### EntityType

Type : MQLONG - entrée

Type d'entité. Type d'entité spécifié par *EntityName*. Il doit s'agir de l'une des valeurs suivantes:

#### MQZAET\_PRINCIPAL

Entité.

#### GRUPE MQZAET\_GROUP

.

#### ObjectName

Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet auquel l'accès doit être extrait. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT\_Q\_MGR, ce nom est identique à *QMgrName*.

### **ObjectType**

Type : MQLONG - entrée

Type d'objet. Type d'entité spécifié par *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

#### **INFO MQOT\_AUTH\_INFO**

Informations d'authentification.

#### **CANAL\_MQTON**

Canal.

#### **MQOT\_CLNTCONN\_CHANNEL**

Canal de connexion client.

#### **MQOT\_PROGRAMME d'écoute**

Programme d'écoute.

#### **MQOT\_NAMELIST**

NAMELIST.

#### **PROCESSUS MQ**

Définition de processus.

#### **MQOT\_Q**

File d'attente.

#### **MQOT\_Q\_DIR**

Gestionnaire de files d'attente.

#### **SERVICE MQOT**

Service.

#### **MQOT\_TOPIC**

:NONE.

### **Droits d'accès**

Type : MQLONG - entrée

Autorité de l'entité. Si l'entité dispose d'un droit d'accès, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO\_\*). S'il possède plusieurs droits, cette zone correspond au OU bit à bit des constantes MQZAO\_\* correspondantes.

### **ComponentData**

Type: MQBYTE xComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

### **Continuation**

Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

#### **MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_GET\_AUTHORITY, cela a le même effet que MQZCI\_CONTINUE.

#### **MQZCI\_CONTINUER**

Passez au composant suivant.

## **MQZCI\_ARRET**

Ne passez pas au composant suivant.

### **CompCode**

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

### **Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') Non autorisé pour l'accès.

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

#### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

#### **MQRC\_UNKNOWN\_ENTITY**

(2292, X'8F4') Entité inconnue du service.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

## **Appel C**

```
MQZ_GET_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,  
                   ObjectType, &Authority, ComponentData,  
                   &Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48  QMgrName;           /* Queue manager name */  
MQCHAR12  EntityName;        /* Entity name */  
MQLONG    EntityType;        /* Entity type */  
MQCHAR48  ObjectName;       /* Object name */  
MQLONG    ObjectType;       /* Object type */  
MQLONG    Authority;        /* Authority of entity */  
MQBYTE    ComponentData[n]; /* Component data */  
MQLONG    Continuation;     /* Continuation indicator set by  
                             component */  
MQLONG    CompCode;         /* Completion code */  
MQLONG    Reason;          /* Reason code qualifying CompCode */
```

## **MQZ\_GET\_AUTHORITY\_2 -Obtention des droits (étendu)**

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_2 et est démarrée par le gestionnaire de files d'attente pour extraire les droits dont dispose une entité pour accéder à l'objet spécifié.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_GET\_AUTHORITY.

MQZ\_GET\_AUTHORITY\_2 est similaire à MQZ\_GET\_AUTHORITY, mais avec le paramètre **EntityName** remplacé par le paramètre **EntityData**.

## Syntaxe

MQZ\_GET\_AUTHORITY\_2( *QMgrName* , *EntityData* , *EntityType* , *ObjectName* , *ObjectType* , *Authority* , *ComponentData* , *Continuation* , *CompCode* , *Reason* )

## Paramètres

### QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

### EntityData

Type: MQZED-entrée

Données d'entité. Données relatives à l'entité pour laquelle l'autorisation sur l'objet doit être extraite. Pour plus d'informations, voir «MQZED-Descripteur d'entité», à la page 1759.

### EntityType

Type : MQLONG - entrée

Type d'entité. Type d'entité spécifié par *EntityData*. Il doit s'agir de l'une des valeurs suivantes:

#### **MQZAET\_PRINCIPAL**

Entité.

#### **GROUPE MQZAET\_GROUP**

### ObjectName

Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet pour lequel le droit d'entité doit être extrait. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT\_Q\_MGR, ce nom est identique à *QMgrName*.

### ObjectType

Type : MQLONG - entrée

Type d'objet. Type d'entité spécifié par *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

#### **INFO MQOT\_AUTH\_INFO**

Informations d'authentification.

#### **CANAL\_MQTON**

Canal.

#### **MQOT\_CLNTCONN\_CHANNEL**

Canal de connexion client.

#### **MQOT\_PROGRAMME d'écoute**

Programme d'écoute.

#### **MQOT\_NAMELIST**

NAMELIST.

#### **PROCESSUS MQ**

Définition de processus.

**MQOT\_Q**

File d'attente.

**MQOT\_Q\_DIR**

Gestionnaire de files d'attente.

**SERVICE MQOT**

Service.

**MQOT\_TOPIC**

:NONE.

**Droits d'accès**

Type : MQLONG - entrée

Autorité de l'entité. Si l'entité dispose d'un droit d'accès, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO\_\*). S'il possède plusieurs droits, cette zone correspond au OU bit à bit des constantes MQZAO\_\* correspondantes.

**ComponentData**

Type: MQBYTE xComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

**Continuation**

Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

**MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_CHECK\_AUTHORITY, cela a le même effet que MQZCI\_STOP.

**MQZCI\_CONTINUER**

Passez au composant suivant.

**MQZCI\_ARRET**

Ne passez pas au composant suivant.

**CompCode**

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

**MQCC\_OK**

Achèvement réussi.

**MQCC\_FAILED**

Echec de l'appel.

**Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') Non autorisé pour l'accès.

**MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

**MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

**MQRC\_UNKNOWN\_ENTITY**

(2292, X'8F4') Entité inconnue du service.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

**Appel C**

```
MQZ_GET_AUTHORITY_2 (QMgrName, &EntityData, EntityType, ObjectName,
                    ObjectType, &Authority, ComponentData,
                    &Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48  QMgrName;          /* Queue manager name */
MQZED     EntityData;       /* Entity data */
MQLONG    EntityType;      /* Entity type */
MQCHAR48  ObjectName;      /* Object name */
MQLONG    ObjectType;      /* Object type */
MQLONG    Authority;       /* Authority of entity */
MQBYTE    ComponentData[n]; /* Component data */
MQLONG    Continuation;    /* Continuation indicator set by
                             component */
MQLONG    CompCode;        /* Completion code */
MQLONG    Reason;         /* Reason code qualifying CompCode */
```

**MQZ\_GET\_EXPLICIT\_AUTHORITY-Obtention des droits explicites**

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_1 et est démarrée par le gestionnaire de files d'attente pour extraire les droits dont une entité dispose pour accéder à l'objet spécifié, y compris (si l'entité est un principal) les droits détenus par les groupes dont le principal est membre. Les droits des profils génériques sont inclus dans le jeu de droits renvoyé.

Sous AIX and Linux, pour le gestionnaire des droits d'accès aux objets (OAM) IBM MQ intégré, les droits renvoyés sont ceux détenus uniquement par le groupe principal du principal.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_GET\_EXPLICIT\_AUTHORITY.

**Syntaxe**

```
MQZ_GET_EXPLICIT_AUTHORITY( QMgrName , EntityName , EntityType , ObjectName ,
ObjectType , Authority , ComponentData , Continuation , CompCode , Reason )
```

**Paramètres****QMgrName**

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

**EntityName**

Type: MQCHAR12 -entrée

Nom de l'entité. Nom de l'entité pour laquelle l'accès à l'objet doit être extrait. La longueur maximale de la chaîne est de 12 caractères ; si elle est plus courte, elle est remplie à droite avec des blancs. Le nom ne se termine pas par un caractère NULL.

### EntityType

Type : MQLONG - entrée

Type d'entité. Type d'entité spécifié par *EntityName*. Il doit s'agir de l'une des valeurs suivantes:

#### **MQZAET\_PRINCIPAL**

Entité.

#### **GROUPE MQZAET\_GROUP**

### ObjectName

Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet pour lequel le droit d'entité doit être extrait. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT\_Q\_MGR, ce nom est identique à *QMgrName*.

### ObjectType

Type : MQLONG - entrée

Type d'objet. Type d'entité spécifié par *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

#### **INFO MQOT\_AUTH\_INFO**

Informations d'authentification.

#### **CANAL\_MQTON**

Canal.

#### **MQOT\_CLNTCONN\_CHANNEL**

Canal de connexion client.

#### **MQOT\_PROGRAMME d'écoute**

Programme d'écoute.

#### **MQOT\_NAMELIST**

NAMELIST.

#### **PROCESSUS MQ**

Définition de processus.

#### **MQOT\_Q**

File d'attente.

#### **MQOT\_Q\_DIR**

Gestionnaire de files d'attente.

#### **SERVICE MQOT**

Service.

#### **MQOT\_TOPIC**

:NONE.

### Droits d'accès

Type : MQLONG - entrée

Autorité de l'entité. Si l'entité dispose d'un droit d'accès, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO\_\*). S'il possède plusieurs droits, cette zone correspond au OU bit à bit des constantes MQZAO\_\* correspondantes.

### ComponentData

Type: MQBYTE x ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des

fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

### Continuation

Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

#### **MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_GET\_AUTHORITY, cela a le même effet que MQZCI\_CONTINUE.

#### **MQZCI\_CONTINUER**

Passez au composant suivant.

#### **MQZCI\_ARRET**

Ne passez pas au composant suivant.

### CompCode

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

### Motif

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') Non autorisé pour l'accès.

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

#### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

#### **MQRC\_UNKNOWN\_ENTITY**

(2292, X'8F4') Entité inconnue du service.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

## Appel C

```
MQZ_GET_EXPLICIT_AUTHORITY (QMgrName, EntityName, EntityType,  
                             ObjectName, ObjectType, &Authority,  
                             ComponentData, &Continuation,  
                             &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48 QMgrName;          /* Queue manager name */
```



```

MQCHAR12  EntityName;      /* Entity name */
MQLONG    EntityType;     /* Entity type */
MQCHAR48  ObjectName;     /* Object name */
MQLONG    ObjectType;     /* Object type */
MQLONG    Authority;      /* Authority of entity */
MQBYTE    ComponentData[n]; /* Component data */
MQLONG    Continuation;   /* Continuation indicator set by
                           component */
MQLONG    CompCode;       /* Completion code */
MQLONG    Reason;         /* Reason code qualifying CompCode */

```

## MQZ\_GET\_EXPLICIT\_AUTHORITY\_2 -Permet d'obtenir des droits explicites (étendu)

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_2 et est démarrée par le gestionnaire de files d'attente pour extraire les droits dont dispose un groupe nommé pour accéder à un objet spécifié (mais sans les droits supplémentaires du groupe **personne**), ou les droits dont dispose le groupe principal du principal nommé pour accéder à un objet spécifié.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_GET\_EXPLICIT\_AUTHORITY.

MQZ\_GET\_EXPLICIT\_AUTHORITY\_2 est similaire à MQZ\_GET\_EXPLICIT\_AUTHORITY, mais avec le paramètre **EntityName** remplacé par le paramètre **EntityData**.

### Syntaxe

MQZ\_GET\_EXPLICIT\_AUTHORITY\_2( *QMgrName* , *EntityData* , *EntityType* , *ObjectName* , *ObjectType* , *Authority* , *ComponentData* , *Continuation* , *CompCode* , *Reason* )

### Paramètres

#### QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

#### EntityData

Type: MQZED-entrée

Données d'entité. Données relatives à l'entité dont l'autorisation sur l'objet doit être extraite. Pour plus d'informations, voir «MQZED-Descripteur d'entité», à la page 1759.

#### EntityType

Type : MQLONG - entrée

Type d'entité. Type d'entité spécifié par *EntityData*. Il doit s'agir de l'une des valeurs suivantes:

#### **MQZAET\_PRINCIPAL**

Entité.

#### **GROUPE MQZAET\_GROUP**

#### ObjectName

Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet pour lequel le droit d'entité doit être extrait. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT\_Q\_MGR, ce nom est identique à *QMgrName*.

## ObjectType

Type : MQLONG - entrée

Type d'objet. Type d'entité spécifié par *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

### **INFO MQOT\_AUTH\_INFO**

Informations d'authentification.

### **CANAL\_MQTON**

Canal.

### **MQOT\_CLNTCONN\_CHANNEL**

Canal de connexion client.

### **MQOT\_PROGRAMME d'écoute**

Programme d'écoute.

### **MQOT\_NAMELIST**

NAMELIST.

### **PROCESSUS MQ**

Définition de processus.

### **MQOT\_Q**

File d'attente.

### **MQOT\_Q\_DIR**

Gestionnaire de files d'attente.

### **SERVICE MQOT**

Service.

### **MQOT\_TOPIC**

:NONE.

## Droits d'accès

Type : MQLONG - entrée

Autorité de l'entité. Si l'entité dispose d'un droit d'accès, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO\_\*). S'il possède plusieurs droits, cette zone correspond au OU bit à bit des constantes MQZAO\_\* correspondantes.

## ComponentData

Type: MQBYTE xComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

## Continuation

Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

### **MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_CHECK\_AUTHORITY, cela a le même effet que MQZCI\_STOP.

### **MQZCI\_CONTINUER**

Passez au composant suivant.

### **MQZCI\_ARRET**

Ne passez pas au composant suivant.

## CompCode

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

**MQCC\_OK**

Achèvement réussi.

**MQCC\_FAILED**

Echec de l'appel.

**Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') Non autorisé pour l'accès.

**MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

**MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

**MQRC\_UNKNOWN\_ENTITY**

(2292, X'8F4') Entité inconnue du service.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

## Appel C

```
MQZ_GET_EXPLICIT_AUTHORITY_2 (QMgrName, &EntityData, EntityType,  
                               ObjectName, ObjectType, &Authority,  
                               ComponentData, &Continuation,  
                               &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48  QMgrName;           /* Queue manager name */  
MQZED     EntityData;        /* Entity data */  
MQLONG    EntityType;        /* Entity type */  
MQCHAR48  ObjectName;        /* Object name */  
MQLONG    ObjectType;        /* Object type */  
MQLONG    Authority;         /* Authority of entity */  
MQBYTE    ComponentData[n]; /* Component data */  
MQLONG    Continuation;      /* Continuation indicator set by  
                               component */  
MQLONG    CompCode;          /* Completion code */  
MQLONG    Reason;            /* Reason code qualifying CompCode */
```

## MQZ\_INIT\_AUTHORITY-Initialisation du service d'autorisation

Cette fonction est fournie par un composant de service d'autorisation et est démarrée par le gestionnaire de files d'attente lors de la configuration du composant. Il est prévu d'appeler MQZEP afin de fournir des informations au gestionnaire de files d'attente.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_INIT\_AUTHORITY.

### Syntaxe

```
MQZ_INIT_AUTHORITY( Hconfig , Options , QMgrName , ComponentDataLength ,  
ComponentData , Version , CompCode , Reason )
```

## Paramètres

### Hconfig

Type: MQHCONFIG-entrée

Descripteur de configuration. Cette poignée représente le composant particulier en cours d'initialisation. Il doit être utilisé par le composant lors de l'appel du gestionnaire de files d'attente avec la fonction MQZEP.

### Options

Type : MQLONG - entrée

Options d'initialisation. Il doit s'agir de l'une des valeurs suivantes:

#### **MQZIO\_PRIMAIRE**

Initialisation principale.

#### **MQZIO\_SECONDAIRE**

Initialisation secondaire.

### QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

### ComponentDataLongueur

Type : MQLONG - entrée

Longueur des données de composant. Longueur en octets de la zone *ComponentData* . Cette longueur est définie dans les données de configuration du composant.

### ComponentData

Type: MQBYTE x ComponentDataLongueur-entrée/sortie

Données de composant. Il est initialisé à zéro avant d'appeler la fonction d'initialisation principale du composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions (y compris la fonction d'initialisation) fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

### Version

Type: MQLONG-entrée / sortie

Numéro de version. En entrée de la fonction d'initialisation, identifie le numéro de version le plus élevé pris en charge par le gestionnaire de files d'attente. La fonction d'initialisation doit modifier, si nécessaire, la version de l'interface qu'elle prend en charge. Si, à son retour, le gestionnaire de files d'attente ne prend pas en charge la version renvoyée par le composant, il appelle la fonction MQZ\_TERM\_AUTHORITY du composant et n'utilise plus ce composant.

Les valeurs suivantes sont prises en charge :

#### **MQZAS\_VERSION\_1**

Version 1.

#### **MQZAS\_VERSION\_2**

Version 2.

#### **MQZAS\_VERSION\_3**

Version 3.

## **MQZAS\_VERSION\_4**

Version 4.

## **MQZAS\_VERSION\_5**

Version 5.

## **MQZAS\_VERSION\_6**

IBM WebSphere MQ 6.

### **CompCode**

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

### **Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_INITIALIZATION\_FAILED**

(2286, X'8EE') L'initialisation a échoué pour une raison non définie.

#### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

## **Appel C**

```
MQZ_INIT_AUTHORITY (Hconfig, Options, QMgrName, ComponentDataLength,  
                    ComponentData, &Version, &CompCode,  
                    &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQHCONFIG  Hconfig;           /* Configuration handle */  
MQLONG     Options;          /* Initialization options */  
MQCHAR48   QMgrName;        /* Queue manager name */  
MQLONG     ComponentDataLength; /* Length of component data */  
MQBYTE     ComponentData[n]; /* Component data */  
MQLONG     Version;         /* Version number */  
MQLONG     CompCode;        /* Completion code */  
MQLONG     Reason;         /* Reason code qualifying CompCode */
```

## **MQZ\_INQUIRE-Demande du service d'autorisation**

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_5 et est démarrée par le gestionnaire de files d'attente pour interroger la fonctionnalité prise en charge.

Lorsque plusieurs composants de service sont utilisés, les composants de service sont appelés dans l'ordre inverse de l'ordre dans lequel ils ont été installés.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_INQUIRE.

## Syntaxe

`MQZ_INQUIRE( QMgrName , SelectorCount , Selectors , IntAttrCount , IntAttrs , CharAttrLength , CharAttrs , SelectorReturned , ComponentData , Continuation , CompCode , Reason )`

## Paramètres

### QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

### SelectorCount

Type : MQLONG - entrée

Nombre de sélecteurs. Nombre de sélecteurs fournis dans le paramètre **Selectors** .

La valeur doit être comprise entre 0 et 256.

### Sélecteurs

Type: MQLONGxSelectorNombre-entrée

Tableau de sélecteurs. Chaque sélecteur identifie un attribut obligatoire et doit être l'un des suivants:

- MQIACF\_INTERFACE\_VERSION (entier)
- MQIACF\_USER\_ID\_SUPPORT (entier)
- MQCACF\_SERVICE\_COMPONENT (caractère)

Les sélecteurs peuvent être spécifiés dans n'importe quel ordre. Le nombre de sélecteurs dans le tableau est indiqué par le paramètre **SelectorCount** .

Les attributs entiers identifiés par les sélecteurs sont renvoyés dans le paramètre **IntAttrs** dans l'ordre dans lequel ils apparaissent dans *Selectors* .

Les attributs de caractères identifiés par les sélecteurs sont renvoyés dans le paramètre **CharAttrs** dans l'ordre dans lequel ils apparaissent *Selectors* .

### IntAttrCount

Type : MQLONG - entrée

Nombre d'attributs de type entier fournis dans le paramètre IntAttrs .

La valeur doit être comprise entre 0 et 256.

### IntAttrs

Type: MQLONG x IntAttrCount-output

Attributs de type entier. Tableau d'attributs de type entier. Les attributs d'entier sont renvoyés dans le même ordre que les sélecteurs d'entier correspondants dans le tableau *Selectors* .

### Nombre CharAttr

Type : MQLONG - entrée

Longueur de la mémoire tampon des attributs de caractères. Longueur en octets du paramètre **CharAttrs** .

La valeur doit être au moins égale à la somme des longueurs des attributs de caractères demandés. Si aucun attribut de caractère n'est demandé, zéro est une valeur valide.

### CharAttrs

Type: MQLONG x CharAttrNombre-sortie

Mémoire tampon des attributs de caractères. Mémoire tampon contenant des attributs de caractères, concaténés ensemble. Les attributs de caractères sont renvoyés dans le même ordre que les sélecteurs de caractères correspondants dans le tableau *Selectors* .

La longueur de la mémoire tampon est indiquée par le paramètre CharAttrCount.

### **SelectorReturned**

Type: MQLONG x SelectorCount -entrée

Sélecteur renvoyé. Tableau de valeurs identifiant les attributs qui ont été renvoyés à partir de l'ensemble demandé par les sélecteurs dans le paramètre Sélecteurs. Le nombre de valeurs de ce tableau est indiqué par le paramètre **SelectorCount** . Chaque valeur du tableau est liée au sélecteur à partir de la position correspondante dans le tableau Sélecteurs. Chaque valeur est l'une des suivantes:

#### **MQZSL\_RENVOYE**

L'attribut demandé par le sélecteur correspondant dans le paramètre **Selectors** a été renvoyé.

#### **MQZSL\_NOT\_RETURNED**

L'attribut demandé par le sélecteur correspondant dans le paramètre **Selectors** n'a pas été renvoyé.

Le tableau est initialisé avec toutes les valeurs *MQZSL\_NOT\_RETURNED*. Lorsqu'un composant de service d'autorisation renvoie un attribut, il définit la valeur appropriée dans le tableau sur *MQZSL\_NOT\_RETURNED*. Cela permet à tout autre composant de service d'autorisation, auquel l'appel d'interrogation est envoyé, d'identifier les attributs qui ont déjà été renvoyés.

### **ComponentData**

Type: MQBYTE x ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

### **Continuation**

Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

#### **MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_CHECK\_AUTHORITY, cela a le même effet que MQZCI\_STOP.

#### **MQZCI\_ARRET**

Ne passez pas au composant suivant.

### **CompCode**

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_WARNING**

Achèvement partiel.

#### **MQCC\_FAILED**

Echec de l'appel.

### **Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_WARNING:

**MQRC\_CHAR\_ATTRS\_TOO\_SHORT**

Espace insuffisant pour les attributs de caractères.

**MQRC\_INT\_COUNT\_TOO\_SMALL**

Espace insuffisant pour les attributs de type entier.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_SELECTOR\_COUNT\_ERREUR**

Le nombre de sélecteurs n'est pas valide.

**MQRC\_SELECTOR\_ERREUR**

Sélecteur d'attribut non valide.

**MQRC\_SELECTOR\_LIMIT\_EXCEEDED**

Trop de sélecteurs spécifiés.

**MQRC\_INT\_ATTR\_COUNT\_ERROR**

Le nombre d'attributs de type entier n'est pas valide.

**MQRC\_INT\_ATTRS\_ARRAY\_ERREUR**

Tableau d'attributs d'entier non valide.

**MQRC\_CHAR\_ATTR\_LENGTH\_ERROR**

Le nombre d'attributs de caractères n'est pas valide.

**MQRC\_CHAR\_ATTRS\_ERREUR**

La chaîne d'attributs de caractères n'est pas valide.

**MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

## Appel C

```
MQZ_INQUIRE (QMgrName, SelectorCount, Selectors, IntAttrCount,  
             &IntAttrs, CharAttrLength, &CharAttrs,  
             SelectorReturned, ComponentData, &Continuation,  
             &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48  QMgrName;           /* Queue manager name */  
MQLONG    SelectorCount;     /* Selector count */  
MQLONG    Selectors[n];      /* Selectors */  
MQLONG    IntAttrCount;     /* IntAttrs count */  
MQLONG    IntAttrs[n];      /* Integer attributes */  
MQLONG    CharAttrCount;    /* CharAttrs count */  
MQLONG    CharAttrs[n];     /* Character attributes */  
MQLONG    SelectorReturned[n]; /* Selector returned */  
MQBYTE    ComponentData[n]; /* Component data */  
MQLONG    Continuation;     /* Continuation indicator set by  
                             component */  
MQLONG    CompCode;         /* Completion code */  
MQLONG    Reason;          /* Reason code qualifying CompCode */
```

## MQZ\_REFRESH\_CACHE-Actualise toutes les autorisations

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_3 et est appelée par le gestionnaire de files d'attente pour actualiser la liste des autorisations détenues en interne par le composant.



L'identificateur de cette fonction (pour MQZEP) est MQZID\_REFRESH\_CACHE (8L).

## Syntaxe

MQZ\_REFRESH\_CACHE( QMgrName , ComponentData , Continuation , CompCode , Reason )

## Paramètres

### QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

### ComponentData

Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors du prochain appel de l'une des fonctions de ce composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

### Continuation

Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

#### MQZCI\_DEFAULT

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_CHECK\_AUTHORITY, cela a le même effet que MQZCI\_STOP.

#### MQZCI\_CONTINUER

Passez au composant suivant.

#### MQZCI\_ARRET

Ne passez pas au composant suivant.

### CompCode

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

#### MQCC\_OK

Achèvement réussi.

#### MQCC\_FAILED

Echec de l'appel.

### Motif

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### MQRC\_NONE

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_WARNING:

## MQRC\_SERVICE\_ERROR

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

## Appel C

```
MQZ_REFRESH_CACHE (QMgrName, ComponentData,  
                  &Continuation, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQCHAR48  QMgrName;          /* Queue manager name */  
MQBYTE    ComponentData[n]; /* Component data */  
MQLONG    Continuation;     /* Continuation indicator set by  
                             component */  
MQLONG    CompCode;        /* Completion code */  
MQLONG    Reason;         /* Reason code qualifying CompCode */
```

## MQZ\_SET\_AUTHORITY-Définition des droits

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_1 et est démarrée par le gestionnaire de files d'attente pour définir les droits d'accès d'une entité à l'objet spécifié.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_SET\_AUTHORITY.

**Remarque :** Cette fonction remplace les droits existants. Pour conserver les droits existants, vous devez les définir à nouveau à l'aide de cette fonction.

## Syntaxe

```
MQZ_SET_AUTHORITY( QMgrName , EntityName , EntityType , ObjectName ,  
ObjectType , Authority , ComponentData , Continuation , CompCode , Reason )
```

## Paramètres

### QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

### EntityName

Type: MQCHAR12 -entrée

Nom de l'entité. Nom de l'entité pour laquelle l'accès à l'objet doit être extrait. La longueur maximale de la chaîne est de 12 caractères ; si elle est plus courte, elle est remplie à droite avec des blancs. Le nom ne se termine pas par un caractère NULL.

### EntityType

Type : MQLONG - entrée

Type d'entité. Type d'entité spécifié par *EntityName*. Il doit s'agir de l'une des valeurs suivantes:

#### MQZAET\_PRINCIPAL

Entité.

#### GROUPE MQZAET\_GROUP

**ObjectName**

Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet auquel l'accès est requis. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT\_Q\_MGR, ce nom est identique à *QMgrName*.

**ObjectType**

Type : MQLONG - entrée

Type d'objet. Type d'entité spécifié par *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

**INFO MQOT\_AUTH\_INFO**

Informations d'authentification.

**CANAL\_MQTON**

Canal.

**MQOT\_CLNTCONN\_CHANNEL**

Canal de connexion client.

**MQOT\_PROGRAMME d'écoute**

Programme d'écoute.

**MQOT\_NAMELIST**

NAMELIST.

**PROCESSUS MQ**

Définition de processus.

**MQOT\_Q**

File d'attente.

**MQOT\_Q\_DIR**

Gestionnaire de files d'attente.

**SERVICE MQOT**

Service.

**MQOT\_TOPIC**

:NONE.

**Droits d'accès**

Type : MQLONG - entrée

Autorité de l'entité. Si un droit est défini, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO\_\*). Si plusieurs droits sont définis, cette zone correspond au OU bit à bit des constantes MQZAO\_\* correspondantes.

**ComponentDataname>**

Type: MQBYTEComponentDataLength -entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

**Continuation**

Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

**MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_GET\_AUTHORITY, cela a le même effet que MQZCI\_CONTINUE.

### **MQZCI\_CONTINUER**

Passez au composant suivant.

### **MQZCI\_ARRET**

Ne passez pas au composant suivant.

#### **CompCode**

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

#### **Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') Non autorisé pour l'accès.

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

#### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

#### **MQRC\_UNKNOWN\_ENTITY**

(2292, X'8F4') Entité inconnue du service.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

## **Appel C**

```
MQZ_SET_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,  
                  ObjectType, Authority, ComponentData,  
                  &Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48  QMgrName;           /* Queue manager name */  
MQCHAR12  EntityName;        /* Entity name */  
MQLONG    EntityType;        /* Entity type */  
MQCHAR48  ObjectName;        /* Object name */  
MQLONG    ObjectType;        /* Object type */  
MQLONG    Authority;         /* Authority to be checked */  
MQBYTE    ComponentData[n]; /* Component data */  
MQLONG    Continuation;      /* Continuation indicator set by  
                             component */  
MQLONG    CompCode;          /* Completion code */  
MQLONG    Reason;           /* Reason code qualifying CompCode */
```

## **MQZ\_SET\_AUTHORITY\_2 -Définition des droits (étendu)**

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_2 et est démarrée par le gestionnaire de files d'attente pour définir les droits d'accès d'une entité à l'objet spécifié.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_SET\_AUTHORITY.

**Remarque :** Cette fonction remplace les droits existants. Pour conserver les droits existants, vous devez les définir à nouveau à l'aide de cette fonction.

MQZ\_SET\_AUTHORITY\_2 est similaire à MQZ\_SET\_AUTHORITY, mais avec le paramètre **EntityName** remplacé par le paramètre **EntityData**.

## Syntaxe

MQZ\_SET\_AUTHORITY\_2( *QMgrName* , *EntityData* , *EntityType* , *ObjectName* , *ObjectType* , *Authority* , *ComponentData* , *Continuation* , *CompCode* , *Reason* )

## Paramètres

### QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

### EntityData

Type: MQZED-entrée

Données d'entité. Données relatives à l'entité dont l'autorisation sur l'objet doit être définie. Pour plus d'informations, voir «MQZED-Descripteur d'entité», à la page 1759.

### EntityType

Type : MQLONG - entrée

Type d'entité. Type d'entité spécifié par *EntityData*. Il doit s'agir de l'une des valeurs suivantes:

#### **MQZAET\_PRINCIPAL**

Entité.

#### **GROUPE MQZAET\_GROUP**

### ObjectName

Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet sur lequel les droits d'entité doivent être définis. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT\_Q\_MGR, ce nom est identique à *QMgrName*.

### ObjectType

Type : MQLONG - entrée

Type d'objet. Type d'entité spécifié par *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

#### **INFO MQOT\_AUTH\_INFO**

Informations d'authentification.

#### **CANAL\_MQTON**

Canal.

#### **MQOT\_CLNTCONN\_CHANNEL**

Canal de connexion client.

#### **MQOT\_PROGRAMME d'écoute**

Programme d'écoute.

**MQOT\_NAMELIST**

NAMELIST.

**PROCESSUS MQ**

Définition de processus.

**MQOT\_Q**

File d'attente.

**MQOT\_Q\_DIR**

Gestionnaire de files d'attente.

**SERVICE MQOT**

Service.

**MQOT\_TOPIC**

:NONE.

**Droits d'accès**

Type : MQLONG - entrée

Autorité de l'entité. Si un droit est défini, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO\_\*). Si plusieurs droits sont définis, cette zone correspond au OU bit à bit des constantes MQZAO\_\* correspondantes.

**ComponentData**

Type: MQBYTE xComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

**Continuation**

Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

**MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_CHECK\_AUTHORITY, cela a le même effet que MQZCI\_STOP.

**MQZCI\_CONTINUER**

Passez au composant suivant.

**MQZCI\_ARRET**

Ne passez pas au composant suivant.

**CompCode**

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

**MQCC\_OK**

Achèvement réussi.

**MQCC\_FAILED**

Echec de l'appel.

**Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') Non autorisé pour l'accès.

**MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

**MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

**MQRC\_UNKNOWN\_ENTITY**

(2292, X'8F4') Entité inconnue du service.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

## Appel C

```
MQZ_SET_AUTHORITY_2 (QMgrName, &EntityData, EntityType, ObjectName,
                    ObjectType, Authority, ComponentData,
                    &Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48  QMgrName;           /* Queue manager name */
MQZED     EntityData;        /* Entity data */
MQLONG    EntityType;       /* Entity type */
MQCHAR48  ObjectName;       /* Object name */
MQLONG    ObjectType;       /* Object type */
MQLONG    Authority;        /* Authority to be checked */
MQBYTE    ComponentData[n]; /* Component data */
MQLONG    Continuation;     /* Continuation indicator set by
                             component */
MQLONG    CompCode;        /* Completion code */
MQLONG    Reason;         /* Reason code qualifying CompCode */
```

## MQZ\_TERM\_AUTHORITY-Arrêt du service d'autorisation

Cette fonction est fournie par un composant de service d'autorisation et est démarrée par le gestionnaire de files d'attente lorsqu'il n'a plus besoin des services de ce composant. La fonction doit effectuer tout nettoyage requis par le composant.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_TERM\_AUTHORITY.

### Syntaxe

```
MQZ_TERM_AUTHORITY( Hconfig , Options , QMgrName , ComponentData , CompCode ,
                    Reason )
```

### Paramètres

**Hconfig**

Type: MQHCONFIG-entrée

Descripteur de configuration. Cette poignée représente le composant particulier en cours d'arrêt. Il doit être utilisé par le composant lors de l'appel du gestionnaire de files d'attente avec la fonction MQZEP.

**Options**

Type : MQLONG - entrée

Options de fin. Il doit s'agir de l'une des valeurs suivantes:

**MQZTO\_PRIMAIRE**

Arrêt principal.

**MQZTO\_SECONDAIRE**

Arrêt secondaire.

**QMgrName**

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

**ComponentData**

Type: MQBYTE x ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre de longueur ComponentDatade l'appel MQZ\_INIT\_AUTHORITY.

Lorsque l'appel MQZ\_TERM\_AUTHORITY est terminé, le gestionnaire de files d'attente supprime ces données.

**CompCode**

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

**MQCC\_OK**

Achèvement réussi.

**MQCC\_FAILED**

Echec de l'appel.

**Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

**Echec de la commande MQRC\_TERMINATION\_FAILED**

(2287, X'8FF') L'arrêt a échoué pour une raison non définie.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

**Appel C**

```
MQZ_TERM_AUTHORITY (Hconfig, Options, QMgrName, ComponentData,  
&CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:



```

MQHCONFIG  Hconfig;           /* Configuration handle */
MQLONG     Options;         /* Termination options */
MQCHAR48   QMgrName;       /* Queue manager name */
MQBYTE     ComponentData[n]; /* Component data */
MQLONG     CompCode;       /* Completion code */
MQLONG     Reason;        /* Reason code qualifying CompCode */

```

## MQZ\_DELETE\_NAME-Nom de suppression

Cette fonction est fournie par un composant de service annuaire et démarrée par le gestionnaire de files d'attente pour supprimer une entrée de la file d'attente spécifiée.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_DELETE\_NAME.

### Syntaxe

MQZ\_DELETE\_NAME( QMgrName , QName , ComponentData , Continuation , CompCode , Reason )

### Paramètres

#### QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

#### QName

Type: MQCHAR48 -entrée

Nom de file d'attente. Nom de la file d'attente pour laquelle un poste doit être supprimé. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

#### ComponentData

Type: MQBYTE x ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre de longueur ComponentDatade l'appel MQZ\_INIT\_NAME.

#### Continuation

Type : MQLONG - sortie

Indicateur de continuation défini par composant. Il doit s'agir de l'une des valeurs suivantes:

#### MQZCI\_DEFAULT

Continuation dépendant du gestionnaire de files d'attente.

#### MQZCI\_ARRET

Ne passez pas au composant suivant.

Pour la commande **MQZ\_DELETE\_NAME** , le gestionnaire de files d'attente ne tente pas de démarrer un autre composant, peu importe ce qui est renvoyé dans le paramètre **Continuation** .

#### CompCode

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

**MQCC\_OK**

Achèvement réussi.

**MQCC\_WARNING**

Avertissement (achèvement partiel).

**MQCC\_FAILED**

Echec de l'appel.

**Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_WARNING:

**MQRC\_UNKNOWN\_NAME**

(2288, X'8F0') Nom de file d'attente introuvable.

**Remarque :** Il se peut qu'il ne soit pas possible de renvoyer ce code si le service sous-jacent répond avec succès dans ce cas.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

**MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

## Appel C

```
MQZ_DELETE_NAME (QMgrName, QName, ComponentData, &Continuation,
                 &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48 QMgrName;          /* Queue manager name */
MQCHAR48 QName;             /* Queue name */
MQBYTE   ComponentData[n]; /* Component data */
MQLONG   Continuation;     /* Continuation indicator set by
                           component */
MQLONG   CompCode;         /* Completion code */
MQLONG   Reason;          /* Reason code qualifying CompCode */
```

## MQZ\_INIT\_NAME-Initialisation du service annuaire

Cette fonction est fournie par un composant de service annuaire et est démarrée par le gestionnaire de files d'attente lors de la configuration du composant. Il est prévu d'appeler MQZEP afin de fournir des informations au gestionnaire de files d'attente.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_INIT\_NAME.

### Syntaxe

```
MQZ_INIT_NAME( Hconfig , Options , QMgrName , ComponentDataLength ,
               ComponentData , Version , CompCode , Reason )
```

## Paramètres

### Hconfig

Type: MQHCONFIG-entrée

Descripteur de configuration. Cette poignée représente le composant particulier en cours d'initialisation. Il doit être utilisé par le composant lors de l'appel du gestionnaire de files d'attente avec la fonction MQZEP.

### Options

Type : MQLONG - entrée

Options d'initialisation. Il doit s'agir de l'une des valeurs suivantes:

#### **MQZIO\_PRIMAIRE**

Initialisation principale.

#### **MQZIO\_SECONDAIRE**

Initialisation secondaire.

### QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

### ComponentDataLongueur

Type : MQLONG - entrée

Longueur des données de composant. Longueur en octets de la zone *ComponentData* . Cette longueur est définie dans les données de configuration du composant.

### ComponentData

Type: MQBYTE x ComponentDataLongueur-entrée/sortie

Données de composant. Il est initialisé à zéro avant d'appeler la fonction d'initialisation principale du composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions (y compris la fonction d'initialisation) fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

### Version

Type: MQLONG-entrée / sortie

Numéro de version. En entrée de la fonction d'initialisation, identifie le numéro de version le plus élevé pris en charge par le gestionnaire de files d'attente. La fonction d'initialisation doit modifier, si nécessaire, la version de l'interface qu'elle prend en charge. Si le gestionnaire de files d'attente ne prend pas en charge la version renvoyée par le composant, il appelle la fonction MQZ\_TERM\_NAME du composant et n'utilise plus ce composant.

Les valeurs suivantes sont prises en charge :

#### **MQZAS\_VERSION\_1**

Version 1.

### CompCode

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

**MQCC\_OK**

Achèvement réussi.

**MQCC\_FAILED**

Echec de l'appel.

**Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_INITIALIZATION\_FAILED**

(2286, X'8EE') L'initialisation a échoué pour une raison non définie.

**MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

**Appel C**

```
MQZ_INIT_NAME (Hconfig, Options, QMgrName, ComponentDataLength,
               ComponentData, &Version, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQHCONFIG  Hconfig;           /* Configuration handle */
MQLONG     Options;          /* Initialization options */
MQCHAR48   QMgrName;        /* Queue manager name */
MQLONG     ComponentDataLength; /* Length of component data */
MQBYTE     ComponentData[n]; /* Component data */
MQLONG     Version;         /* Version number */
MQLONG     CompCode;        /* Completion code */
MQLONG     Reason;         /* Reason code qualifying CompCode */
```

**MQZ\_INSERT\_NAME-Nom d'insertion**

Cette fonction est fournie par un composant de service annuaire et est démarrée par le gestionnaire de files d'attente pour insérer une entrée pour la file d'attente spécifiée, contenant le nom du gestionnaire de files d'attente propriétaire de la file d'attente. Si la file d'attente est déjà définie dans le service, l'appel échoue.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_INSERT\_NAME.

**Syntaxe**

```
MQZ_INSERT_NAME( QMgrName , QName , ResolvedQMgrName , ComponentData ,
                 Continuation , CompCode , Reason )
```

**Paramètres****QMgrName**

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

### **QName**

Type: MQCHAR48 -entrée

Nom de file d'attente. Nom de la file d'attente pour laquelle un poste doit être inséré. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

### **ResolvedQMgrNom**

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente résolu. Nom du gestionnaire de files d'attente dans lequel la file d'attente est résolue. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

### **ComponentData**

Type: MQBYTE xComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions (y compris la fonction d'initialisation) fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_NAME.

### **Continuation**

Type: MQLONG-entrée / sortie

Indicateur de continuation défini par composant. Pour MQZ\_INSERT\_NAME, le gestionnaire de files d'attente ne tente pas de démarrer un autre composant, quel qu'il soit renvoyé dans le paramètre **Continuation**.

Les valeurs suivantes sont prises en charge :

#### **MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

#### **MQZCI\_ARRET**

Ne passez pas au composant suivant.

### **CompCode**

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

### **Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_Q\_ALREADY\_EXISTS**

(2290, X'8F2') L'objet de file d'attente existe déjà.

## **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

## **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

## **Appel C**

```
MQZ_INSERT_NAME (QMgrName, QName, ResolvedQMgrName, ComponentData,  
&Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48 QMgrName;          /* Queue manager name */  
MQCHAR48 QName;            /* Queue name */  
MQCHAR48 ResolvedQMgrName; /* Resolved queue manager name */  
MQBYTE ComponentData[n];  /* Component data */  
MQLONG Continuation;      /* Continuation indicator set by  
                           component */  
MQLONG CompCode;          /* Completion code */  
MQLONG Reason;            /* Reason code qualifying CompCode */
```

## **MQZ\_LOOKUP\_NAME-Nom de recherche**

Cette fonction est fournie par un composant de service annuaire et démarrée par le gestionnaire de files d'attente pour extraire le nom du gestionnaire de files d'attente propriétaire, pour une file d'attente spécifiée.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_LOOKUP\_NAME.

### **Syntaxe**

```
MQZ_LOOKUP_NAME( QMgrName , QName , ResolvedQMgrName , ComponentData ,  
Continuation , CompCode , Reason )
```

### **Paramètres**

#### **QMgrName**

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

#### **QName**

Type: MQCHAR48 -entrée

Nom de file d'attente. Nom de la file d'attente pour laquelle un poste doit être résolu. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

#### **ResolvedQMgrNom**

Type: MQCHAR48 -sortie

Nom du gestionnaire de files d'attente résolu. Si la fonction aboutit, il s'agit du nom du gestionnaire de files d'attente propriétaire de la file d'attente.

Le nom renvoyé par le composant de service doit être rempli à droite avec des blancs sur toute la longueur du paramètre ; le nom ne doit pas se terminer par un caractère NULL ou contenir des blancs de début ou imbriqués.

### **ComponentData**

Type: MQBYTEExComponentDataLength -entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions (y compris la fonction d'initialisation) fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_NAME.

### **Continuation**

Type : MQLONG - sortie

Indicateur de continuation défini par composant. Pour MQZ\_LOOKUP\_NAME, le gestionnaire de files d'attente indique s'il doit démarrer un autre composant de service annuaire, comme suit:

- Si *CompCode* est défini sur MQCC\_OK, aucun autre composant n'est démarré, quelle que soit la valeur renvoyée dans *Continuation*.
- Si *CompCode* n'est pas MQCC\_OK, un autre composant est démarré, sauf si *Continuation* est MQZCI\_STOP.

Les valeurs suivantes sont prises en charge :

#### **MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

#### **MQZCI\_CONTINUER**

Passez au composant suivant.

#### **MQZCI\_ARRET**

Ne passez pas au composant suivant.

### **CompCode**

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

### **Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

#### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

#### **MQRC\_UNKNOWN\_NOM\_QQ**

(2288, X'8F0') Nom de file d'attente introuvable.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

## Appel C

```
MQZ_LOOKUP_NAME (QMgrName, QName, ResolvedQMgrName, ComponentData,  
&Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48 QMgrName;          /* Queue manager name */  
MQCHAR48 QName;            /* Queue name */  
MQCHAR48 ResolvedQMgrName; /* Resolved queue manager name */  
MQBYTE ComponentData[n];  /* Component data */  
MQLONG Continuation;      /* Continuation indicator set by  
                           component */  
MQLONG CompCode;          /* Completion code */  
MQLONG Reason;            /* Reason code qualifying CompCode */
```

## MQZ\_TERM\_NAME-Arrêt du service de nom

Cette fonction est fournie par un composant de service annuaire et est démarrée par le gestionnaire de files d'attente lorsqu'il n'a plus besoin des services de ce composant. La fonction doit effectuer tout nettoyage requis par le composant.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_TERM\_NAME.

### Syntaxe

```
MQZ_TERM_NAME( Hconfig , Options , QMgrName , ComponentData , CompCode ,  
Reason )
```

### Paramètres

#### Hconfig

Type: MQHCONFIG-entrée

Descripteur de configuration. Cette poignée représente le composant particulier en cours d'arrêt. Il est utilisé par le composant lors de l'appel du gestionnaire de files d'attente avec la fonction MQZEP.

#### Options

Type : MQLONG - entrée

Options de fin. Il doit s'agir de l'une des valeurs suivantes:

##### **MQZTO\_PRIMAIRE**

Arrêt principal.

##### **MQZTO\_SECONDAIRE**

Arrêt secondaire.

#### QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

#### ComponentData

Type: MQBYTE x ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions (y compris la fonction d'initialisation) fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.



Les données de composant sont dans la mémoire partagée accessible à tous les processus.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_NAME.

Lorsque l'appel MQZ\_TERM\_NAME est terminé, le gestionnaire de files d'attente supprime ces données.

### CompCode

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

#### MQCC\_OK

Achèvement réussi.

#### MQCC\_FAILED

Echec de l'appel.

### Motif

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### MQRC\_NONE

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### Echec de la commande MQRC\_TERMINATION\_FAILED

(2287, X'8FF') L'arrêt a échoué pour une raison non définie.

#### MQRC\_SERVICE\_NOT\_AVAILABLE

(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

## Appel C

```
MQZ_TERM_NAME (Hconfig, Options, QMgrName, ComponentData, &CompCode,  
&Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQHCONFIG Hconfig; /* Configuration handle */  
MQLONG Options; /* Termination options */  
MQCHAR48 QMgrName; /* Queue manager name */  
MQBYTE ComponentData[n]; /* Component data */  
MQLONG CompCode; /* Completion code */  
MQLONG Reason; /* Reason code qualifying CompCode */
```

## MQZAC-Contexte d'application

La structure MQZAC est utilisée dans l'appel MQZ\_AUTHENTICATE\_USER pour le paramètre *ApplicationContext*. Ce paramètre spécifie les données relatives à l'application appelante.

[Le tableau 1](#) récapitule les zones de la structure.

Tableau 838. Zones dans MQZAC	
Zone	Description
<u>StrucId</u>	Identificateur de structure
<u>Version</u>	Numéro de version de structure

Tableau 838. Zones dans MQZAC (suite)

<b>Zone</b>	<b>Description</b>
<u>ProcessId</u>	Identificateur de processus
<u>ThreadId</u>	Identificateur d'unité d'exécution
<u>ApplName</u>	Nom d'application
<u>UserID</u>	ID utilisateur
<u>IDEffectiveUser</u>	ID utilisateur effectif
<u>Environnement</u>	Environnement
<u>CallerType</u>	Type d'appelant
<u>AuthenticationType</u>	Type d'authentification
<u>BindType</u>	Type de liaison

## Zones

### StrucId

Type: MQCHAR4 -entrée

Identificateur de structure. La valeur est la suivante:

#### **ID\_STRUC\_MQZAC**

Identificateur de la structure de contexte d'application.

Pour le langage de programmation C, la constante MQZAC\_STRUC\_ID\_ARRAY est également définie ; elle a la même valeur que MQZAC\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### Version

Type : MQLONG - entrée

Numéro de version de la structure. La valeur est la suivante:

#### **MQZAC\_VERSION\_1**

Structure de contexte d'application Version-1 . La constante MQZAC\_CURRENT\_VERSION indique le numéro de version de la version en cours.

### ProcessId

Type: MQPID-entrée

Identificateur de processus de l'application.

### ThreadId

Type: MQTID-entrée

Identificateur d'unité d'exécution de l'application.

### ApplName

Type: MQCHAR28 -entrée

Nom de l'application.

### UserID

Type: MQCHAR12 -entrée

ID utilisateur. Sous AIX and Linux , cette zone indique l'ID utilisateur réel de l'application. Sous Windows , cette zone indique l'ID utilisateur de l'application.

### ID EffectiveUser

Type: MQCHAR12 -entrée

Identificateur utilisateur effectif. Sous AIX and Linux , cette zone indique l'ID utilisateur effectif de l'application. Sous Windows , cette zone est vide.

### Environnement

Type : MQLONG - entrée

Environnement. Cette zone indique l'environnement à partir duquel l'appel a été effectué. La zone est l'une des valeurs suivantes:

#### **SERVEUR\_COMMAND\_MQ**

Serveur de commandes

#### **MQXE\_MQSC**

Interpréteur de commandes **runmqsc**

#### **MQXE\_MCA**

Agent MCA MQXE\_OTHER

#### **MQXE\_AUTRES**

Environnement non défini

### CallerType

Type : MQLONG - entrée

Type d'appelant. Cette zone indique le type de programme qui a effectué l'appel. La zone est l'une des valeurs suivantes:

#### **MQXACT\_EXTERNAL**

L'appel est externe au gestionnaire de files d'attente.

#### **MQXACT\_INTERNAL**

L'appel est interne au gestionnaire de files d'attente.

### AuthenticationType

Type : MQLONG - entrée

Type d'authentification. Cette zone indique le type d'authentification en cours d'exécution. La zone est l'une des valeurs suivantes:

#### **MQZAT\_INITIAL\_CONTEXT**

L'appel d'authentification est dû à l'initialisation du contexte utilisateur. Cette valeur est utilisée lors d'un appel MQCONN ou MQCONNX.

#### **MQZAT\_CHANGE\_CONTEXT**

L'appel d'authentification est dû à la modification du contexte de l'utilisateur. Cette valeur est utilisée lorsque l'agent MCA modifie le contexte utilisateur. Rubrique parent: MQZAC-

### BindType

Type : MQLONG - entrée

Type de liaison. Cette zone indique le type de liaison en cours d'utilisation. La zone est l'une des valeurs suivantes:

#### **MQCNO\_FASTPATH\_BINDING**

Liaison Fastpath.

#### **MQCNO\_LIEN\_PARTAGE**

Liaison partagée.

#### **MQCNO\_LIEN\_ISOLÉ\_LIAISON**

Liaison isolée.

### Déclaration C

Déclarez les zones de la structure comme suit:

```
typedef struct tagMQZAC MQZAC;  
struct tagMQZAC {  
    MQCHAR4   StructId;           /* Structure identifier */  
    MQLONG    Version;           /* Structure version number */
```

```

MQPID      ProcessId;          /* Process identifier */
MQTID      ThreadId;         /* Thread identifier */
MQCHAR28   ApplName;        /* Application name */
MQCHAR12   UserID;          /* User identifier */
MQCHAR12   EffectiveUserID; /* Effective user identifier */
MQLONG     Environment;     /* Environment */
MQLONG     CallerType;      /* Caller type */
MQLONG     AuthenticationType; /* Authentication type */
MQLONG     BindType;        /* Bind type */
};

```

## MQZAD-Données de droits d'accès

La structure MQZAD est utilisée dans l'appel MQZ\_ENUMERATE\_AUTHORITY\_DATA pour deux paramètres, une entrée et une sortie.

Pour plus d'informations sur les paramètres **Filter** et **AuthorityBuffer**, voir «MQZ\_ENUMERATE\_AUTHORITY\_DATA-Énumération des données de droits d'accès», à la page 1717 :

- MQZAD est utilisé pour le paramètre **Filter** qui est l'entrée de l'appel. Ce paramètre indique les critères de sélection à utiliser pour sélectionner les données de droits d'accès renvoyées par l'appel.
- MQZAD est également utilisé pour le paramètre **AuthorityBuffer** qui est généré à partir de l'appel. Ce paramètre indique les autorisations pour une combinaison de nom de profil, de type d'objet et d'entité.

Tableau 1. récapitule les zones de la structure.

Tableau 839. Zones dans MQZAD	
Zone	Description
<u>StrucId</u>	Identificateur de structure
<u>Version</u>	Numéro de version de structure
<u>ProfileName</u>	Nom de profil
<u>ObjectType</u>	Type d'objet
<u>Droits d'accès</u>	Droits d'accès
<u>EntityDataPtr</u>	Pointeur vers les données d'entité
<u>EntityType</u>	Type d'entité
<u>Options</u>	Options

## Zones

### StrucId

Type: MQCHAR4 -entrée

Identificateur de structure. La valeur est la suivante:

#### ID\_STRUC\_MQZ

Identificateur de la structure de données de droits d'accès.

Pour le langage de programmation C, la constante MQZAD\_STRUC\_ID\_ARRAY est également définie ; elle a la même valeur que MQZAD\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### Version

Type : MQLONG - entrée

Numéro de version de la structure. La valeur est la suivante:

### **MQZAD\_VERSION\_1**

Structure de contexte d'application Version-1 . La constante MQZAD\_CURRENT\_VERSION indique le numéro de version de la version en cours.

La constante suivante indique le numéro de version de la version en cours:

### **MQZAD\_CURRENT\_VERSION**

Version actuelle de la structure de données de droits d'accès.

### **ProfileName**

Type: MQCHAR48 -entrée

Nom du profil.

Pour le paramètre **Filter** , cette zone correspond au nom de profil pour lequel des données de droits d'accès sont requises. Si le nom est entièrement à blanc jusqu'à la fin de la zone ou jusqu'au premier caractère NULL, les données de droits pour tous les noms de profil sont renvoyées.

Pour le paramètre **AuthorityBuffer** , cette zone correspond au nom d'un profil qui correspond aux critères de sélection spécifiés.

### **ObjectType**

Type : MQLONG - entrée

Type d'objet.

Pour le paramètre **Filter** , cette zone correspond au type d'objet pour lequel des données de droits d'accès sont requises. Si la valeur est MQOT\_ALL, les données de droits d'accès pour tous les types d'objet sont renvoyées.

Pour le paramètre **AuthorityBuffer** , cette zone correspond au type d'objet auquel s'applique le profil identifié par le paramètre **ProfileName** .

La valeur est l'une des suivantes ; pour le paramètre **Filter** , la valeur MQOT\_ALL est également valide:

### **INFO MQOT\_AUTH\_INFO**

Informations d'authentification

### **CANAL\_MQTON**

Canal

### **MQOT\_CLNTCONN\_CHANNEL**

Canal de connexion client

### **MQOT\_PROGRAMME d'écoute**

Programme d'écoute

### **MQOT\_NAMELIST**

Liste de noms

### **PROCESSUS MQ**

Définition de processus

### **MQOT\_Q**

File d'attente

### **MQOT\_Q\_DIR**

Gestionnaire de files d'attente

### **SERVICE MQOT**

Service

### **Droits d'accès**

Type : MQLONG - entrée

Droits d'accès.

Pour le paramètre **Filter** , cette zone est ignorée.

Pour le paramètre **AuthorityBuffer** , cette zone représente les autorisations dont dispose l'entité pour les objets identifiés par **ProfileName** et **ObjectType**. Si l'entité ne possède qu'un seul droit d'accès, la zone est égale à la valeur d'autorisation appropriée (constante MQZAO\_ \*). Si l'entité possède plusieurs droits d'accès, la zone est le OU bit à bit des constantes MQZAO\_ \* correspondantes.

### EntityDataPtr

Type: PMQZED-entrée

Adresse de la structure MQZED identifiant une entité.

Pour le paramètre **Filter** , cette zone pointe vers une structure MQZED qui identifie l'entité pour laquelle des données de droits d'accès sont requises. Si **EntityDataPtr** est le pointeur null, les données de droits d'accès de toutes les entités sont renvoyées.

Pour le paramètre **AuthorityBuffer** , cette zone pointe vers une structure MQZED qui identifie l'entité pour laquelle des données de droits d'accès ont été renvoyées.

### EntityType

Type : MQLONG - entrée

Type d'entité.

Pour le paramètre **Filter** , cette zone indique le type d'entité pour lequel des données de droits d'accès sont requises. Si la valeur est MQZAET\_NONE, les données de droits d'accès pour tous les types d'entité sont renvoyées.

Pour le paramètre **AuthorityBuffer** , cette zone indique le type de l'entité identifiée par la structure MQZED désignée par le paramètre **EntityDataPtr** .

La valeur est l'une des suivantes ; pour le paramètre **Filter** , la valeur MQZAET\_NONE est également valide:

#### **MQZAET\_PRINCIPAL**

Principale

#### **GROUPE MQZAET\_GROUP**

Groupe

### Options

Type: MQAUTHOPT-entrée

Options. Cette zone indique les options qui permettent de contrôler les profils affichés. L'une des valeurs suivantes doit être spécifiée:

#### **MQAUTHOPT\_NOM\_ALL\_MATCHING**

Affiche tous les profils

#### **MQAUTHOPT\_NOM\_EXPLICITE**

Affiche les profils ayant exactement le même nom que celui spécifié dans la zone **ProfileName** .

En outre, l'un des éléments suivants doit également être spécifié:

#### **MQAUTHOPT\_ENTITY\_SET**

Affiche tous les profils utilisés pour calculer les droits cumulés de l'entité sur l'objet spécifié par le paramètre **ProfileName** . Le paramètre **ProfileName** ne doit pas contenir de caractères génériques.

- Si l'entité spécifiée est un principal, pour chaque membre de l'ensemble {entity, groups} , le profil le plus applicable qui s'applique à l'objet est affiché.
- Si l'entité spécifiée est un groupe, le profil le plus applicable du groupe qui s'applique à l'objet est affiché.
- Si cette valeur est spécifiée, les valeurs de **ProfileName**, **ObjectType**, **EntityType** et le nom d'entité spécifié dans la structure **EntityDataPtr** MQZED doivent toutes être non vides.

Si vous avez spécifié MQAUTHOPT\_NAME\_ALL\_MATCHING, vous pouvez également spécifier la valeur suivante:

## MQAUTHOPT\_ENTITY\_EXPLICIT

Affiche les profils ayant exactement le même nom d'entité que le nom d'entité spécifié dans la structure **EntityDataPtr** MQZED.

## Déclaration C

```
typedef struct tagMQZAD MQZAD;
struct tagMQZAD {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;          /* Structure version number */
    MQCHAR48   ProfileName;      /* Profile name */
    MQLONG     ObjectType;       /* Object type */
    MQLONG     Authority;        /* Authority */
    PMQZED     EntityDataPtr;    /* Address of MQZED structure identifying an
                                entity */
    MQLONG     EntityType;       /* Entity type */
    MQAUTHOPT  Options;         /* Options */
};
```

## MQZED-Descripteur d'entité

La structure MQZED est utilisée dans un certain nombre d'appels de service d'autorisation pour spécifier l'entité pour laquelle l'autorisation doit être vérifiée.

Tableau 1. récapitule les zones de la structure.

Tableau 840. Zones dans MQZED	
Zone	Description
<u>StrucId</u>	Identificateur de structure
<u>Version</u>	Version
<u>EntityName Ptr</u>	Nom de l'entité
<u>EntityDomainPtr</u>	Pointeur de domaine d'entité
<u>SecurityId</u>	Identificateur de sécurité
<u>CorrelationPtr</u>	Pointeur de corrélation

## Zones

### StrucId

Type: MQCHAR4 -entrée

Identificateur de structure. La valeur est la suivante:

#### ID\_STRUC\_MQZ

Identificateur de la structure de descripteur d'entité.

Pour le langage de programmation C, la constante MQZED\_STRUC\_ID\_ARRAY est également définie ; elle a la même valeur que MQZED\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

### Version

Type : MQLONG - entrée

Numéro de version de la structure. La valeur est la suivante:

#### MQZED\_VERSION\_1

Structure de descripteur d'entité Version-1 .

La constante suivante indique le numéro de version de la version en cours:

#### VERSION\_MQZED\_CURRENT\_VERSION

Version actuelle de la structure de descripteur d'entité.

### EntityNamePtr

Type: PMQCHAR-input

Nom du profil.

Adresse du nom de l'entité. Il s'agit d'un pointeur vers le nom de l'entité dont l'autorisation doit être vérifiée.

### EntityDomainPtr

Type: PMQCHAR-input

Adresse du nom de domaine de l'entité. Il s'agit d'un pointeur vers le nom du domaine contenant la définition de l'entité dont l'autorisation doit être vérifiée.

### SecurityId

Type: MQBYTE40 -entrée

Droits d'accès.

Identificateur de sécurité. Il s'agit de l'identificateur de sécurité dont l'autorisation doit être vérifiée.

### CorrelationPtr

Type: MQPTR-entrée

Pointeur de corrélation. Cela facilite la transmission de données corrélationnelles entre la fonction d'authentification de l'utilisateur et d'autres fonctions OAM appropriées.

## Déclaration C

```
typedef struct tagMQZED MQZED;
struct tagMQZED {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;          /* Structure version number */
    PMQCHAR    EntityNamePtr;    /* Address of entity name */
    PMQCHAR    EntityDomainPtr; /* Address of entity domain name */
    MQBYTE40   SecurityId;       /* Security identifier */
    MQPTR      CorrelationPtr;   /* Address of correlation data */
}
```

## MQZEP-Ajout d'un point d'entrée de composant

Un composant de service démarre cette fonction, lors de l'initialisation, pour ajouter un point d'entrée au vecteur de point d'entrée de ce composant de service.

### Syntaxe

MQZEP ( *Hconfig* , *Fonction* , *EntryPoint* , *CompCode* , *Motif* )

### Paramètres

#### Hconfig

Type: MQHCONFIG-entrée

Descripteur de configuration. Ce descripteur représente le composant en cours de configuration pour ce service installable particulier. Il doit être identique au composant transmis à la fonction de configuration de composant par le gestionnaire de files d'attente lors de l'appel d'initialisation de composant.

#### Fonction

Type : MQLONG - entrée

Identificateur de fonction. Les valeurs valides sont définies pour chaque service installable.

Si MQZEP est appelé plusieurs fois pour la même fonction, le dernier appel effectué fournit le point d'entrée utilisé.



## EntryPoint

Type: PMQFUNC-entrée

Point d'entrée de fonction. Il s'agit de l'adresse du point d'entrée fourni par le composant pour exécuter la fonction.

La valeur NULL est valide et indique que la fonction n'est pas fournie par ce composant. La valeur NULL est prise en compte pour les points d'entrée qui ne sont pas définis à l'aide de MQZEP.

## CompCode

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

### MQCC\_OK

Achèvement réussi.

### MQCC\_FAILED

Echec de l'appel.

## Motif

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

### MQRC\_NONE

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

### Erreur de fonction MQRC\_FUNCTION\_ERROR

(2281, X'8E9') Identificateur de fonction incorrect.

### ERREUR MQRC\_HCONFIG\_ERROR

(2280, X'8E8') Descripteur de configuration non valide.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

## Appel C

```
MQZEP (Hconfig, Function, EntryPoint, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONFIG Hconfig; /* Configuration handle */
MQLONG Function; /* Function identifier */
PMQFUNC EntryPoint; /* Function entry point */
MQLONG CompCode; /* Completion code */
MQLONG Reason; /* Reason code qualifying CompCode */
```

## MQZFP-Paramètres libres

La structure MQZFP est utilisée dans l'appel MQZ\_FREE\_USER pour le paramètre *FreeParms* . Ce paramètre indique les données relatives à la ressource à libérer.

*Tableau 1.* récapitule les zones de la structure.

Tableau 841. Zones dans MQZFP	
Zone	Description
<u>StrucId</u>	Identificateur de structure
<u>Version</u>	Version

Tableau 841. Zones dans MQZFP (suite)

Zone	Description
Reserved	Réservé, zone
CorrelationPtr	Pointeur de corrélation

## Zones

### StrucId

Type: MQCHAR4 -entrée

Identificateur de structure. La valeur est la suivante:

#### ID\_STRUC\_MQZ

Identificateur de la structure de contexte d'identité. Pour le langage de programmation C, la constante MQZIC\_STRUC\_ID\_ARRAY est également définie ; elle a la même valeur que MQZIC\_STRUC\_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

### Version

Type : MQLONG - entrée

Numéro de version de la structure. La valeur est la suivante:

#### MQZFP\_VERSION\_1

Structure des paramètres libres Version-1 .

La constante suivante indique le numéro de version de la version en cours:

#### VERSION\_MQZFP\_CURRENT\_VERSION

Version actuelle de la structure des paramètres libres.

### Réservé

Type: MQBYTE8 -entrée

Zone réservée. La valeur initiale est null.

### CorrelationPtr

Type: MQPTR-entrée

Pointeur de corrélation. Adresse des données de corrélation relatives à la ressource à libérer.

## Déclaration C

```
typedef struct tagMQZFP MQZFP;
struct tagMQZFP {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;          /* Structure version number */
    MQBYTE8    Reserved;        /* Reserved field */
    MQPTR      CorrelationPtr;   /* Address of correlation data */
};
```

## MQZIC-Contexte d'identité

La structure MQZIC est utilisée dans l'appel MQZ\_AUTHENTICATE\_USER pour le paramètre *IdentityContext* .

La structure MQZIC contient des informations de contexte d'identité, qui identifient l'utilisateur de l'application qui a placé le message dans une file d'attente pour la première fois:

- Le gestionnaire de files d'attente remplit la zone *UserIdentifier* avec un nom qui identifie l'utilisateur, la manière dont le gestionnaire de files d'attente peut effectuer cette opération dépend de l'environnement dans lequel l'application s'exécute.

- Le gestionnaire de files d'attente remplit la zone *AccountingToken* avec un jeton ou un numéro qu'il a déterminé à partir de l'application qui a inséré le message.
- Les applications peuvent utiliser la zone *DonnéesApplIdentity* pour toute information supplémentaire qu'elles souhaitent inclure sur l'utilisateur (par exemple, un mot de passe chiffré).

Les applications disposant des droits appropriés peuvent définir le contexte d'identité à l'aide de la fonction MQZ\_AUTHENTICATE\_USER.

Un identificateur de sécurité (SID) Windows est stocké dans la zone *AccountingToken* lorsqu'un message est créé sous IBM MQ for Windows. Le SID peut être utilisé pour compléter la zone *UserIdentifier* et pour établir les données d'identification d'un utilisateur.

*Tableau 1.* récapitule les zones de la structure.

<i>Tableau 842. Zones dans MQZIC</i>	
<b>Zone</b>	<b>Description</b>
<u>StrucId</u>	Identificateur de structure
<u>Version</u>	Version
<u>UserIdentifier</u>	ID utilisateur
<u>AccountingToken</u>	Jeton de comptabilité
<u>ApplIdentityData</u>	Données sur l'identité de l'application

## Zones

### StrucId

Type: MQCHAR4 -entrée

Identificateur de structure. La valeur est la suivante:

#### **ID\_STRUC\_MQZ**

Identificateur de la structure de contexte d'identité. Pour le langage de programmation C, la constante MQZIC\_STRUC\_ID\_ARRAY est également définie ; elle a la même valeur que MQZIC\_STRUC\_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

### Version

Type : MQLONG - entrée

Numéro de version de la structure. La valeur est la suivante:

#### **MQZIC\_VERSION\_1**

Structure de contexte d'identité Version-1 .

La constante suivante indique le numéro de version de la version en cours:

#### **MQZIC\_CURRENT\_VERSION**

Version actuelle de la structure de contexte d'identité.

### UserIdentifier

Type: MQCHAR12 -entrée

ID utilisateur. Il fait partie du contexte d'identité du message. *UserIdentifier* indique l'ID utilisateur de l'application à l'origine du message. Le gestionnaire de files d'attente traite ces informations comme des données de type caractères, mais ne définit pas leur format. Pour plus d'informations sur la zone *UserIdentifier* , voir «*UserIdentifier (MQCHAR12) pour MQMD*», à la page 477.

### AccountingToken

Type: MQBYTE32 -entrée

Jeton de comptabilité. Il fait partie du contexte d'identité du message. *AccountingToken* permet à une application de provoquer une facturation appropriée du travail effectué à la suite du message. Le gestionnaire de files d'attente traite ces informations comme une chaîne de bits et ne vérifie pas leur

contenu. Pour plus d'informations sur la zone *AccountingToken*, voir «[AccountingToken \(MQBYTE32 pour MQMD](#)», à la page 478.

### ApplIdentityData

Type: MQCHAR32 -entrée

Données d'application relatives à l'identité. Il fait partie du contexte d'identité du message. ApplIdentityLes données sont des informations définies par la suite d'applications qui peuvent être utilisées pour fournir des informations supplémentaires sur l'origine du message. Par exemple, elle peut être définie par des applications s'exécutant avec des droits utilisateur appropriés pour indiquer si les données d'identité sont dignes de confiance. Pour plus d'informations sur la zone de données ApplIdentity, voir «[Données ApplIdentity\(MQCHAR32\) pour MQMD](#)», à la page 480.

## Déclaration C

```
typedef struct tagMQZED MQZED;
struct tagMQZED {
    MQCHAR4    StructId;           /* Structure identifier */
    MQLONG     Version;           /* Structure version number */
    MQCHAR12   UserIdentifier;     /* User identifier */
    MQBYTE32   AccountingToken;   /* Accounting token */
    MQCHAR32   ApplIdentityData;  /* Application data relating to identity */
};
```

## IBM i Informations de référence de l'interface des services installables sur IBM i

Utilisez ces informations pour comprendre les informations de référence des services installables pour IBM i.

Pour chaque fonction, il existe une description, y compris l'identificateur de fonction (pour MQZEP).

Les *paramètres* sont répertoriés dans l'ordre dans lequel ils doivent apparaître. Ils doivent tous être présents.

Chaque nom de paramètre est suivi de son type de données entre parenthèses. Il s'agit des types de données élémentaires décrits dans «[Types de données élémentaires](#)», à la page 1044.

L'appel de langage C est également fourni, après la description des paramètres.

### Concepts associés

**IBM i** [Services et composants installables pour IBM i](#)

**ALW** [Services et composants installables pour UNIX, Linux et Windows](#)

### Référence associée

«[Informations de référence de l'interface des services installables](#)», à la page 1699

Cette collection de rubriques fournit des informations de référence pour les services installables.

## IBM i MQZEP (Ajouter un point d'entrée de composant) sur IBM i

Cette fonction est appelée par un composant de service, lors de l'initialisation, pour ajouter un point d'entrée au vecteur de point d'entrée de ce composant de service.

## Syntaxe

```
MQZEP (Hconfig, Function, EntryPoint, CompCode, Reason)
```

## Paramètres

L'appel MQZEP comporte les paramètres suivants.

### **Hconfig (MQHCONFIG)-entrée**

Descripteur de configuration.

Cet indicateur représente le composant en cours de configuration pour ce service installable particulier. Il doit être identique à celui transmis à la fonction de configuration de composant par le gestionnaire de files d'attente lors de l'appel d'initialisation de composant.

### **Fonction (MQLONG)-entrée**

Identificateur de fonction.

Les valeurs valides sont définies pour chaque service installable. Si MQZEP est appelé plusieurs fois pour la même fonction, le dernier appel effectué fournit le point d'entrée qui est utilisé.

### **EntryPoint (PMQFUNC)-entrée**

Point d'entrée de fonction.

Il s'agit de l'adresse du point d'entrée fourni par le composant pour exécuter la fonction. La valeur NULL est valide et indique que la fonction n'est pas fournie par ce composant. La valeur NULL est supposée pour les points d'entrée qui ne sont pas définis à l'aide de MQZEP.

### **CompCode (MQLONG)-sortie**

Code achèvement.

Il peut s'agir de :

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

### **Cause (MQLONG)-sortie**

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### **Erreur de fonction MQRC\_FUNCTION\_ERROR**

(2281, X'8E9') Identificateur de fonction incorrect.

#### **ERREUR MQRC\_HCONFIG\_ERROR**

(2280, X'8E8') Descripteur de configuration non valide.

Pour plus d'informations sur ces codes anomalie, voir [Messages et codes anomalie](#).

## **Appel C**

```
MQZEP (Hconfig, Function, EntryPoint, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONFIG Hconfig;      /* Configuration handle */
MQLONG    Function;     /* Function identifier */
PMQFUNC   EntryPoint;   /* Function entry point */
MQLONG    CompCode;     /* Completion code */
MQLONG    Reason;       /* Reason code qualifying CompCode */
```



## **MQHCONFIG (descripteur de configuration) sous IBM i**

Le type de données MQHCONFIG représente un descripteur de configuration, c'est-à-dire le composant configuré pour un service installable particulier. Un descripteur de configuration doit être aligné sur sa limite naturelle.

Les applications doivent tester les variables de ce type uniquement pour l'égalité.

## Déclaration C

```
typedef void MQPOINTER MQHCONFIG;
```

## IBM i **PMQFUNC (Pointer to function) sous IBM i**

Pointeur vers une fonction.

## Déclaration C

```
typedef void MQPOINTER PMQFUNC;
```

## IBM i **MQZ\_AUTHENTICATE\_USER (authentification de l'utilisateur) sur IBM i**

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_5 . Il est appelé par le gestionnaire de files d'attente pour authentifier un utilisateur ou pour définir des zones de contexte d'identité.

Il est appelé lorsqu'un contexte d'application utilisateur IBM MQ est établi. Cela se produit lors des appels de connexion au point où le contexte utilisateur de l'application est initialisé et à chaque point où le contexte utilisateur de l'application est modifié. Chaque fois qu'un appel de connexion est effectué, les informations de contexte utilisateur de l'application sont réacquises dans la zone *IdentityContext* .

L'identificateur de cette fonction (pour MQZEP) est MQZID\_AUTHENTICATE\_USER.

## Syntaxe

**MQZ\_AUTHENTICATE\_USER (QMgrName, SecurityParms, ApplicationContext, IdentityContext, CorrelationPtr, ComponentData, Continuation, CompCode, Reason)**

## Paramètres

L'appel MQZ\_AUTHENTICATE\_USER comporte les paramètres suivants.

### **QMgrName (MQCHAR48)-entrée**

Nom du gestionnaire de files d'attente

Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini. Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

### **SecurityParms (MQCSP)-entrée**

Paramètres de sécurité.

Données relatives à l'ID utilisateur, au mot de passe et au type d'authentification.

Lors d'un appel MQCONN MQI, ce paramètre contient des valeurs null ou des valeurs par défaut.

### **ApplicationContext (MQZAC)-entrée**

Contexte d'application.

Données relatives à l'application appelante. Pour plus d'informations, voir «MQZAC (contexte d'application) sur IBM i», à la page 1796. Lors de chaque appel MQCONN ou MQCONNX MQI, les informations de contexte utilisateur de la structure MQZAC sont réacquises.

### **IdentityContext (MQZIC)-entrée / sortie**

Contexte d'identité.

Lors de l'entrée dans la fonction d'authentification de l'utilisateur, identifie le contexte d'identité en cours. La fonction d'authentification de l'utilisateur peut changer cela, auquel cas le gestionnaire de files d'attente adopte le nouveau contexte d'identité. Pour plus de détails sur la structure MQZIC, voir «MQZIC (contexte d'identité) sur IBM i», à la page 1803 .

### **CorrelationPtr (MQPTR)-sortie**

Pointeur de corrélation.

Indique l'adresse des données de corrélation. Ce pointeur est ensuite transmis à d'autres appels OAM.

### **ComponentData (MQBYTE x ComponentDataLongueur)-entrée / sortie**

Données de composant.

Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant. La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

### **Continuation (MQLONG)-sortie**

Indicateur de continuation.

Les valeurs suivantes peuvent être spécifiées :

#### **MQZCI\_DEFAULT**

Continuation dépendante des autres composants.

#### **MQZCI\_ARRET**

Ne passez pas au composant suivant.

### **CompCode (MQLONG)-sortie**

Code achèvement.

Il peut s'agir de :

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

### **Cause (MQLONG)-sortie**

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

Pour plus d'informations sur ces codes anomalie, voir [Messages et codes anomalie](#).

## **Appel C**

```
MQZ_AUTHENTICATE_USER (QMgrName, SecurityParms, ApplicationContext,  
IdentityContext, &CorrelationPtr, ComponentData,  
&Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```

MQCHAR48  QMgrName;           /* Queue manager name */
MQCSP     SecurityParms;    /* Security parameters */
MQZAC     ApplicationContext; /* Application context */
MQZIC     IdentityContext;  /* Identity context */
MQPTR     CorrelationPtr;   /* Correlation pointer */
MQBYTE    ComponentData[n]; /* Component data */
MQLONG    Continuation;     /* Continuation indicator set by
                             component */
MQLONG    CompCode;         /* Completion code */
MQLONG    Reason;          /* Reason code qualifying CompCode */

```

## IBM i MQZ\_CHECK\_AUTHORITY (Vérification des droits) sur IBM i

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_1 et est appelée par le gestionnaire de files d'attente pour vérifier si une entité est autorisée à effectuer une ou plusieurs actions particulières sur un objet spécifié.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_CHECK\_AUTHORITY.

### Syntaxe

**MQZ\_CHECK\_AUTHORITY** (*QMgrName*, *EntityName*, *EntityType*, *ObjectName*, *ObjectType*, *Authority*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

### Paramètres

L'appel MQZ\_CHECK\_AUTHORITY comporte les paramètres suivants.

#### QMgrName (MQCHAR48)-entrée

Nom du gestionnaire de files d'attente

Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini. Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

#### EntityName (MQCHAR12)-entrée

Nom de l'entité.

Nom de l'entité dont l'autorisation sur l'objet doit être vérifiée. La longueur maximale de la chaîne est de 12 caractères ; si elle est plus courte, elle est remplie à droite avec des blancs. Le nom ne se termine pas par un caractère NULL.

Il n'est pas essentiel que cette entité soit connue du service de sécurité sous-jacent. S'il n'est pas connu, les autorisations du groupe spécial **nobody** (auquel toutes les entités sont supposées appartenir) sont utilisées pour le contrôle. Un nom vide est valide et peut être utilisé de cette manière.

#### EntityType (MQLONG)-entrée

Type d'entité.

Type d'entité spécifié par *EntityName*. Il peut s'agir de :

**MQZAET\_PRINCIPAL**

Entité.

**GROUPE MQZAET\_GROUP**

#### ObjectName (MQCHAR48)-entrée

Nom de l'objet.

Nom de l'objet auquel l'accès est requis. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.



Si *ObjectType* est MQOT\_Q\_MGR, ce nom est identique à *QMgrName*.

### **ObjectType (MQLONG)-entrée**

Type d'objet.

Type d'entité spécifié par *ObjectName*. Il peut s'agir de :

#### **INFO MQOT\_AUTH\_INFO**

Informations d'authentification.

#### **CANAL\_MQTON**

Canal.

#### **MQOT\_CLNTCONN\_CHANNEL**

Canal de connexion client.

#### **MQOT\_PROGRAMME d'écoute**

Programme d'écoute.

#### **MQOT\_NAMELIST**

NAMELIST.

#### **PROCESSUS MQ**

Définition de processus.

#### **MQOT\_Q**

File d'attente.

#### **MQOT\_Q\_DIR**

Gestionnaire de files d'attente.

#### **SERVICE MQOT**

Service.

### **Droits d'accès (MQLONG)-entrée**

Droits à vérifier.

Si une autorisation est vérifiée, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO\_\*). Si plusieurs autorisations sont vérifiées, il s'agit du OU bit à bit des constantes MQZAO\_\* correspondantes.

Les autorisations suivantes s'appliquent à l'utilisation des appels MQI:

#### **MQZAO\_CONNECT**

Possibilité d'utiliser l'appel MQCONN.

#### **MQZAO\_PARCOURIR**

Possibilité d'utiliser l'appel MQGET avec une option de navigation.

Cela permet de spécifier l'option MQGMO\_BROWSE\_FIRST, MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR ou MQGMO\_BROWSE\_NEXT sur l'appel MQGET.

#### **MQZAO\_ENTREE**

Possibilité d'utiliser l'appel MQGET avec une option d'entrée.

Cela permet à l'option MQOO\_INPUT\_SHARED, MQOO\_INPUT\_EXCLUSIVE ou MQOO\_INPUT\_AS\_Q\_DEF d'être spécifiée sur l'appel MQOPEN.

#### **MQZAO\_OUTPUT**

Possibilité d'utiliser l'appel MQPUT.

Cela permet à l'option MQOO\_OUTPUT d'être spécifiée sur l'appel MQOPEN.

#### **MQZAO\_INQUIRE**

Possibilité d'utiliser l'appel MQINQ.

Cela permet de spécifier l'option MQOO\_INQUIRE sur l'appel MQOPEN.

#### **MQZAO\_SET**

Possibilité d'utiliser l'appel MQSET.

Cela permet de spécifier l'option MQOO\_SET sur l'appel MQOPEN.

**MQZAO\_PASS\_IDENTITY\_CONTEXT**

Possibilité de transmettre le contexte d'identité.

Cela permet de spécifier l'option MQOO\_PASS\_IDENTITY\_CONTEXT sur l'appel MQOPEN et l'option MQPMO\_PASS\_IDENTITY\_CONTEXT sur les appels MQPUT et MQPUT1 .

**MQZAO\_PASS\_ALL\_CONTEXT**

Possibilité de transmettre tout le contexte.

Cela permet de spécifier l'option MQOO\_PASS\_ALL\_CONTEXT sur l'appel MQOPEN et l'option MQPMO\_PASS\_ALL\_CONTEXT sur les appels MQPUT et MQPUT1 .

**MQZAO\_SET\_IDENTITY\_CONTEXT**

Possibilité de définir le contexte d'identité.

Cela permet de spécifier l'option MQOO\_SET\_IDENTITY\_CONTEXT sur l'appel MQOPEN et l'option MQPMO\_SET\_IDENTITY\_CONTEXT sur les appels MQPUT et MQPUT1 .

**MQZAO\_SET\_ALL\_CONTEXT**

Possibilité de définir tous les contextes.

Cela permet de spécifier l'option MQOO\_SET\_ALL\_CONTEXT sur l'appel MQOPEN et l'option MQPMO\_SET\_ALL\_CONTEXT sur les appels MQPUT et MQPUT1 .

**MQZAO\_ALTERNATE\_USER\_AUTHORITY**

Possibilité d'utiliser des droits utilisateur de remplacement.

Cela permet de spécifier l'option MQOO\_ALTERNATE\_USER\_AUTHORITY sur l'appel MQOPEN et l'option MQPMO\_ALTERNATE\_USER\_AUTHORITY sur l'appel MQPUT1 .

**MQZAO\_ALL\_MQI**

Toutes les autorisations MQI.

Ceci permet d'activer toutes les autorisations décrites précédemment.

Les autorisations suivantes s'appliquent à l'administration d'un gestionnaire de files d'attente:

**MQZAO\_CREER**

Possibilité de créer des objets d'un type spécifié.

**MQZAO\_DELETE**

Possibilité de supprimer un objet spécifié.

**MQZAO\_DISPLAY**

Possibilité d'afficher les attributs d'un objet spécifié.

**MODIFICATION MQZAO\_DE**

Possibilité de modifier les attributs d'un objet spécifié.

**MQZAO\_CLEAR**

Possibilité de supprimer tous les messages d'une file d'attente spécifiée.

**MQZAO\_AUTORISATION**

Possibilité d'autoriser d'autres utilisateurs pour un objet spécifié.

**CONTROLE MQZ**

Possibilité de démarrer, d'arrêter ou de lancer une commande ping sur un objet de canal non client.

**MQZAO\_CONTRÔLE\_ÉTENDU**

Possibilité de réinitialiser un numéro de séquence ou de résoudre un message en attente de validation sur un objet canal non client.

**MQZAO\_ALL\_ADMIN**

Toutes les autorisations d'administration, autres que MQZAO\_CREATE.

Les autorisations suivantes s'appliquent à l'utilisation de l'interface MQI et à l'administration d'un gestionnaire de files d'attente:

**MQZAO\_ALL**

Toutes les autorisations, autres que MQZAO\_CREATE.

## **MQZAO\_AUCUN**

Aucune autorisation.

### **ComponentData (MQBYTE x ComponentDataLongueur)-entrée / sortie**

Données de composant.

Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors du prochain appel de l'une des fonctions de ce composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

### **Continuation (MQLONG)-sortie**

Indicateur de continuation défini par composant.

Les valeurs suivantes peuvent être spécifiées :

#### **MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_CHECK\_AUTHORITY, cela a le même effet que MQZCI\_STOP.

#### **MQZCI\_CONTINUER**

Passez au composant suivant.

#### **MQZCI\_ARRET**

Ne passez pas au composant suivant.

### **CompCode (MQLONG)-sortie**

Code achèvement.

Il peut s'agir de :

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

### **Cause (MQLONG)-sortie**

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') Non autorisé pour l'accès.

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

#### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir [Messages et codes anomalie](#).

## **Appel C**

```
MQZ_CHECK_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,  
                    ObjectType, Authority, ComponentData,  
                    &Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```

MQCHAR48  QMgrName;          /* Queue manager name */
MQCHAR12  EntityName;      /* Entity name */
MQLONG    EntityType;      /* Entity type */
MQCHAR48  ObjectName;     /* Object name */
MQLONG    ObjectType;     /* Object type */
MQLONG    Authority;      /* Authority to be checked */
MQBYTE    ComponentData[n]; /* Component data */
MQLONG    Continuation;   /* Continuation indicator set by
                           component */
MQLONG    CompCode;       /* Completion code */
MQLONG    Reason;        /* Reason code qualifying CompCode */

```

## MQZ\_CHECK\_PRIVILEGED-vérifie si l'utilisateur est privilégié

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_6 et est appelée par le gestionnaire de files d'attente pour déterminer si un utilisateur spécifié est un utilisateur privilégié.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_CHECK\_PRIVILEGED.

### Syntaxe

```

MQZ_CHECK_PRIVILEGED( QMgrName , EntityData , EntityType , ComponentData ,
Continuation , CompCode , Reason )

```

### Paramètres

#### QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

#### EntityData

Type: MQZED-entrée

Données d'entité. Données relatives à l'entité à vérifier. Pour plus d'informations, voir «MQZED-Descripteur d'entité», à la page 1759.

#### EntityType

Type : MQLONG - entrée

Type d'entité. Type d'entité spécifié par EntityData. Il doit s'agir de l'une des valeurs suivantes:

#### **MQZAET\_PRINCIPAL**

Entité.

#### **GROUPE MQZAET\_GROUP**

#### ComponentData

Type: MQBYTEExComponentDataLength -entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

#### Continuation

Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

#### **MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_CHECK\_AUTHORITY, cela a le même effet que MQZCI\_STOP.

#### **MQZCI\_CONTINUER**

Passez au composant suivant.

#### **MQZCI\_ARRET**

Ne passez pas au composant suivant.

Si l'appel à un composant échoue (c'est-à-dire que *CompCode* renvoie MQCC\_FAILED) et que le paramètre *Continuation* est MQZCI\_DEFAULT ou MQZCI\_CONTINUE, le gestionnaire de files d'attente continue d'appeler d'autres composants s'il en existe.

Si l'appel aboutit (c'est-à-dire si *CompCode* renvoie MQCC\_OK), aucun autre composant n'est appelé, quelle que soit la valeur de *Continuation*.

Si l'appel échoue et que le paramètre *Continuation* est MQZCI\_STOP, aucun autre composant n'est appelé et l'erreur est renvoyée au gestionnaire de files d'attente. Les composants ne connaissant pas les appels précédents, le paramètre *Continuation* est toujours défini sur MQZCI\_DEFAULT avant l'appel.

#### **CompCode**

Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

#### **Motif**

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_NOT\_PRIVILEGED**

(2584, X'A18') Cet utilisateur n'est pas un ID utilisateur privilégié.

#### **MQRC\_UNKNOWN\_ENTITY**

(2292, X'8F4') Entité inconnue du service.

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

#### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir [Codes anomalie et achèvement d'API](#).

## **Appel C**

```
MQZ_CHECK_PRIVILEGED (QMgrName, &EntityData, EntityType,  
                    ComponentData, &Continuation,  
                    &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```

MQCHAR48  QMgrName;          /* Queue manager name */
MQZED     EntityData;      /* Entity name */
MQLONG    EntityType;      /* Entity type */
MQBYTE    ComponentData[n]; /* Component data */
MQLONG    Continuation;    /* Continuation indicator set by
                             component */
MQLONG    CompCode;        /* Completion code */
MQLONG    Reason;          /* Reason code qualifying CompCode */

```



## MQZ\_COPY\_ALL\_AUTHORITY (Copie de tous les droits) sur IBM i

Cette fonction est fournie par un composant de service d'autorisation. Il est appelé par le gestionnaire de files d'attente pour copier toutes les autorisations actuellement en vigueur pour un objet de référence dans un autre objet.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_COPY\_ALL\_AUTHORITY.

### Syntaxe

**MQZ\_COPY\_ALL\_AUTHORITY** (*QMgrName, RefObjectName, ObjectName, ObjectType, ComponentData, Continuation, CompCode, Reason*)

### Paramètres

L'appel MQZ\_COPY\_ALL\_AUTHORITY comporte les paramètres suivants.

#### QMgrName (MQCHAR48)-entrée

Nom du gestionnaire de files d'attente

Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

#### RefObjectName (MQCHAR48)-entrée

Nom de l'objet de référence.

Nom de l'objet de référence dont les autorisations doivent être copiées. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

#### ObjectName (MQCHAR48)-entrée

Nom de l'objet.

Nom de l'objet pour lequel les accès doivent être définis. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

#### ObjectType (MQLONG)-entrée

Type d'objet.

Type d'objet spécifié par *RefObjectName* et *ObjectName*. Il peut s'agir de :

##### INFO MQOT\_AUTH\_INFO

Informations d'authentification.

##### CANAL\_MQTON

Canal.

##### MQOT\_CLNTCONN\_CHANNEL

Canal de connexion client.

##### MQOT\_PROGRAMME d'écoute

Programme d'écoute.

**MQOT\_NAMELIST**

NAMELIST.

**PROCESSUS MQ**

Définition de processus.

**MQOT\_Q**

File d'attente.

**MQOT\_Q\_DIR**

Gestionnaire de files d'attente.

**SERVICE MQOT**

Service.

**ComponentData (MQBYTE x ComponentDataLongueur)-entrée / sortie**

Données de composant.

Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors du prochain appel de l'une des fonctions de ce composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

**Continuation (MQLONG)-sortie**

Indicateur de continuation défini par composant.

Les valeurs suivantes peuvent être spécifiées :

**MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_COPY\_ALL\_AUTHORITY, cela a le même effet que MQZCI\_STOP.

**MQZCI\_CONTINUER**

Passez au composant suivant.

**MQZCI\_ARRET**

Ne passez pas au composant suivant.

**CompCode (MQLONG)-sortie**

Code achèvement.

Il peut s'agir de :

**MQCC\_OK**

Achèvement réussi.

**MQCC\_FAILED**

Echec de l'appel.

**Cause (MQLONG)-sortie**

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

**MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

**MQRC\_UNKNOWN\_REF\_OBJECT**

(2294, X'8F6') Objet de référence inconnu.

Pour plus d'informations sur ces codes anomalie, voir [Messages et codes anomalie](#).

## Appel C

```
MQZ_COPY_ALL_AUTHORITY (QMgrName, RefObjectName, ObjectName, ObjectType,  
                        ComponentData, &Continuation, &CompCode,  
                        &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48  QMgrName;           /* Queue manager name */  
MQCHAR48  RefObjectName;      /* Reference object name */  
MQCHAR48  ObjectName;        /* Object name */  
MQLONG    ObjectType;        /* Object type */  
MQBYTE    ComponentData[n];  /* Component data */  
MQLONG    Continuation;      /* Continuation indicator set by  
                             component */  
MQLONG    CompCode;          /* Completion code */  
MQLONG    Reason;            /* Reason code qualifying CompCode */
```

## MQZ\_DELETE\_AUTHORITY (droits de suppression) sur IBM i

Cette fonction est fournie par un composant de service d'autorisation et est appelée par le gestionnaire de files d'attente pour supprimer toutes les autorisations associées à l'objet spécifié.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_DELETE\_AUTHORITY.

### Syntaxe

```
MQZ_DELETE_AUTHORITY (QMgrName, ObjectName, ObjectType,  
                       ComponentData, Continuation, CompCode, Reason)
```

### Paramètres

L'appel MQZ\_DELETE\_AUTHORITY comporte les paramètres suivants.

#### **QMgrName (MQCHAR48)-entrée**

Nom du gestionnaire de files d'attente

Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

#### **ObjectName (MQCHAR48)-entrée**

Nom de l'objet.

Nom de l'objet pour lequel les accès doivent être supprimés. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT\_Q\_MGR, ce nom est identique à *QMgrName*.

#### **ObjectType (MQLONG)-entrée**

Type d'objet.

Type d'entité spécifié par *ObjectName*. Il peut s'agir de :

##### **INFO MQOT\_AUTH\_INFO**

Informations d'authentification.

##### **CANAL\_MQTON**

Canal.



**MQOT\_CLNTCONN\_CHANNEL**

Canal de connexion client.

**MQOT\_PROGRAMME d'écoute**

Programme d'écoute.

**MQOT\_NAMELIST**

NAMELIST.

**PROCESSUS MQ**

Définition de processus.

**MQOT\_Q**

File d'attente.

**MQOT\_Q\_DIR**

Gestionnaire de files d'attente.

**SERVICE MQOT**

Service.

**ComponentData (MQBYTE x ComponentDataLongueur)-entrée / sortie**

Données de composant.

Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors du prochain appel de l'une des fonctions de ce composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

**Continuation (MQLONG)-sortie**

Indicateur de continuation défini par composant.

Les valeurs suivantes peuvent être spécifiées :

**MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_DELETE\_AUTHORITY, cela a le même effet que MQZCI\_STOP.

**MQZCI\_CONTINUER**

Passez au composant suivant.

**MQZCI\_ARRET**

Ne passez pas au composant suivant.

**CompCode (MQLONG)-sortie**

Code achèvement.

Il peut s'agir de :

**MQCC\_OK**

Achèvement réussi.

**MQCC\_FAILED**

Echec de l'appel.

**Cause (MQLONG)-sortie**

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

## **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir [Messages et codes anomalie](#).

## **Appel C**

```
MQZ_DELETE_AUTHORITY (QMgrName, ObjectName, ObjectType, ComponentData,  
&Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48 QMgrName;          /* Queue manager name */  
MQCHAR48 ObjectName;       /* Object name */  
MQLONG   ObjectType;       /* Object type */  
MQBYTE   ComponentData[n]; /* Component data */  
MQLONG   Continuation;     /* Continuation indicator set by  
                           component */  
MQLONG   CompCode;        /* Completion code */  
MQLONG   Reason;         /* Reason code qualifying CompCode */
```

## **IBM i MQZ\_ENUMERATE\_AUTHORITY\_DATA (Énumération des données de droits d'accès) sur IBM i**

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_4 et est appelée à plusieurs reprises par le gestionnaire de files d'attente pour extraire toutes les données de droits d'accès qui correspondent aux critères de sélection spécifiés lors du premier appel.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_ENUMERATE\_AUTHORITY\_DATA.

## **Syntaxe**

```
MQZ_ENUMERATE_AUTHORITY_DATA (QMgrName, StartEnumeration,  
Filter, AuthorityBufferLength, AuthorityBuffer, AuthorityDataLength,  
ComponentData, Continuation, CompCode, Reason)
```

## **Paramètres**

L'appel MQZ\_ENUMERATE\_AUTHORITY\_DATA comporte les paramètres suivants.

### **QMgrName (MQCHAR48)-entrée**

Nom du gestionnaire de files d'attente

Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

### **StartEnumeration (MQLONG)-entrée**

Indicateur signalant si l'appel doit commencer l'énumération.

Indique si l'appel doit démarrer l'énumération des données de droits d'accès ou continuer l'énumération des données de droits d'accès lancée par un appel précédent à MQZ\_ENUMERATE\_AUTHORITY\_DATA. La valeur est l'une des suivantes :

#### **DEMARRAGE MQZ**

Démarrer l'énumération.

L'appel est appelé avec cette valeur pour démarrer l'énumération des données de droits d'accès. Le paramètre **Filter** indique les critères de sélection à utiliser pour sélectionner les données de droits d'accès renvoyées par cet appel et les appels successifs.

## **MQZSE\_CONTINUE**

Continuez l'énumération.

L'appel est appelé avec cette valeur pour continuer l'énumération des données de droits d'accès. Le paramètre **Filter** est ignoré dans ce cas et peut être spécifié en tant que pointeur null (les critères de sélection sont déterminés par le paramètre **Filter** spécifié par l'appel qui a défini *StartEnumeration* sur MQZSE\_START).

## **Filtre (MQZAD)-entrée**

filtrer.

Si *StartEnumeration* est MQZSE\_START, *Filter* indique les critères de sélection à utiliser pour sélectionner les données de droits d'accès à renvoyer. Si *Filter* est le pointeur null, aucun critère de sélection n'est utilisé, c'est-à-dire que toutes les données de droits d'accès sont renvoyées. Pour plus de détails sur les critères de sélection pouvant être utilisés, voir «MQZAD (données de droits d'accès) sur IBM i», à la page 1798 .

Si *StartEnumeration* est MQZSE\_CONTINUE, *Filter* est ignoré et peut être spécifié en tant que pointeur null.

## **AuthorityBufferLength (MQLONG)-entrée**

Longueur de *AuthorityBuffer*.

Il s'agit de la longueur en octets du paramètre **AuthorityBuffer** . La mémoire tampon des droits d'accès doit être suffisamment grande pour contenir les données à renvoyer.

## **AuthorityBuffer (MQZAD)-sortie**

Données de droits d'accès.

Il s'agit de la mémoire tampon dans laquelle les données de droits d'accès sont renvoyées. La mémoire tampon doit être suffisamment grande pour accueillir une structure MQZAD, une structure MQZED, plus le nom d'entité le plus long et le nom de domaine le plus long définis.

**Remarque :** Ce paramètre est défini en tant que MQZAD, car MQZAD se produit toujours au début de la mémoire tampon. Toutefois, si la mémoire tampon est effectivement déclarée en tant que MQZAD, la mémoire tampon sera trop petite-elle doit être plus grande qu'une MQZAD pour pouvoir accueillir les noms d'entité et de domaine MQZAD, MQZED et MQZAD.

## **AuthorityDataLongueur (MQLONG)-sortie**

Longueur des données renvoyées dans *AuthorityBuffer*.

Il s'agit de la longueur des données renvoyées dans *AuthorityBuffer*. Si la mémoire tampon des droits est trop petite, *AuthorityDataLength* est défini sur la longueur de la mémoire tampon requise et l'appel renvoie le code achèvement MQCC\_FAILED et le code anomalie MQRC\_BUFFER\_LENGTH\_ERROR.

## **ComponentData (MQBYTE x ComponentDataLongueur)-entrée / sortie**

Données de composant.

Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors du prochain appel de l'une des fonctions de ce composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

## **Continuation (MQLONG)-sortie**

Indicateur de continuation défini par composant.

Les valeurs suivantes peuvent être spécifiées :

### **MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_ENUMERATE\_AUTHORITY\_DATA, cela a le même effet que MQZCI\_CONTINUE.

## **MQZCI\_CONTINUER**

Passez au composant suivant.

## **MQZCI\_ARRET**

Ne passez pas au composant suivant.

### **CompCode (MQLONG)-sortie**

Code achèvement.

Il peut s'agir de :

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

### **Cause (MQLONG)-sortie**

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

#### **MQRC\_NO\_DATA\_XX\_ENCODE\_CASE\_ONE disponible**

(2379, X'94B') Aucune donnée disponible.

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

Pour plus d'informations sur ces codes anomalie, voir [Messages et codes anomalie](#).

## **Appel C**

```
MQZ_ENUMERATE_AUTHORITY_DATA (QMgrName, StartEnumeration, &Filter,  
                               AuthorityBufferLength,  
                               &AuthorityBuffer,  
                               &AuthorityDataLength, ComponentData,  
                               &Continuation, &CompCode,  
                               &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48  QMgrName;           /* Queue manager name */  
MQLONG    StartEnumeration;   /* Flag indicating whether call should  
                               start enumeration */  
MQZAD     Filter;             /* Filter */  
MQLONG    AuthorityBufferLength; /* Length of AuthorityBuffer */  
MQZAD     AuthorityBuffer;    /* Authority data */  
MQLONG    AuthorityDataLength; /* Length of data returned in  
                               AuthorityBuffer */  
MQBYTE    ComponentData[n];  /* Component data */  
MQLONG    Continuation;      /* Continuation indicator set by  
                               component */  
MQLONG    CompCode;          /* Completion code */  
MQLONG    Reason;            /* Reason code qualifying CompCode */
```

## **MQZ\_FREE\_USER-Utilisateur libre**

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_5 et est appelée par le gestionnaire de files d'attente pour libérer la ressource allouée associée. Elle est appelée lorsqu'une application a fini de s'exécuter dans tous les contextes utilisateur, par exemple lors d'un appel MQI MQDISC.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_FREE\_USER.

## **MQZ\_GET\_AUTHORITY (droits d'accès) sur IBM i**

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_1 et est appelée par le gestionnaire de files d'attente pour extraire les droits d'accès d'une entité à l'objet spécifié.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_GET\_AUTHORITY.

### Syntaxe

**Commande MQZ\_GET\_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName, ObjectType, Authority, ComponentData, Continuation, CompCode, Reason)**

### Paramètres

L'appel MQZ\_GET\_AUTHORITY comporte les paramètres suivants.

#### **QMgrName (MQCHAR48)-entrée**

Nom du gestionnaire de files d'attente

Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

#### **EntityName (MQCHAR12)-entrée**

Nom de l'entité.

Nom de l'entité dont l'accès à l'objet doit être extrait. La longueur maximale de la chaîne est de 12 caractères ; si elle est plus courte, elle est remplie à droite avec des blancs. Le nom ne se termine pas par un caractère NULL.

#### **EntityType (MQLONG)-entrée**

Type d'entité.

Type d'entité spécifié par *EntityName*. La valeur suivante peut être spécifiée:

##### **MQZAET\_PRINCIPAL**

Entité.

##### **GROUPE MQZAET\_GROUP**

.

#### **ObjectName (MQCHAR48)-entrée**

Nom de l'objet.

Nom de l'objet pour lequel les droits de l'entité doivent être extraits. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT\_Q\_MGR, ce nom est identique à *QMgrName*.

#### **ObjectType (MQLONG)-entrée**

Type d'objet.

Type d'entité spécifié par *ObjectName*. Il peut s'agir de :

##### **INFO MQOT\_AUTH\_INFO**

Informations d'authentification.

##### **CANAL\_MQTON**

Canal.

##### **MQOT\_CLNTCONN\_CHANNEL**

Canal de connexion client.

**MQOT\_PROGRAMME d'écoute**

Programme d'écoute.

**MQOT\_NAMELIST**

NAMELIST.

**PROCESSUS MQ**

Définition de processus.

**MQOT\_Q**

File d'attente.

**MQOT\_Q\_DIR**

Gestionnaire de files d'attente.

**SERVICE MQOT**

Service.

**Droits d'accès (MQLONG)-sortie**

Autorité de l'entité.

Si l'entité dispose d'un droit d'accès, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO\_\*). S'il possède plusieurs droits, cette zone correspond au OU bit à bit des constantes MQZAO\_\* correspondantes.

**ComponentData (MQBYTE x ComponentDataLongueur)-entrée / sortie**

Données de composant.

Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors du prochain appel de l'une des fonctions de ce composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

**Continuation (MQLONG)-sortie**

Indicateur de continuation défini par composant.

Les valeurs suivantes peuvent être spécifiées :

**MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_GET\_AUTHORITY, cela a le même effet que MQZCI\_CONTINUE.

**MQZCI\_CONTINUER**

Passez au composant suivant.

**MQZCI\_ARRET**

Ne passez pas au composant suivant.

**CompCode (MQLONG)-sortie**

Code achèvement.

Il peut s'agir de :

**MQCC\_OK**

Achèvement réussi.

**MQCC\_FAILED**

Echec de l'appel.

**Cause (MQLONG)-sortie**

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') Non autorisé pour l'accès.

**MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

**MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

**MQRC\_UNKNOWN\_ENTITY**

(2292, X'8F4') Entité inconnue du service.

Pour plus d'informations sur ces codes anomalie, voir [Messages et codes anomalie](#).

## Appel C

```
MQZ_GET_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,  
                  ObjectType, &Authority, ComponentData,  
                  &Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48 QMgrName;          /* Queue manager name */  
MQCHAR12 EntityName;       /* Entity name */  
MQLONG   EntityType;       /* Entity type */  
MQCHAR48 ObjectName;      /* Object name */  
MQLONG   ObjectType;      /* Object type */  
MQLONG   Authority;       /* Authority of entity */  
MQBYTE   ComponentData[n]; /* Component data */  
MQLONG   Continuation;    /* Continuation indicator set by  
                           component */  
MQLONG   CompCode;        /* Completion code */  
MQLONG   Reason;         /* Reason code qualifying CompCode */
```

## IBM i MQZ\_GET\_EXPLICIT\_AUTHORITY (Obtention des droits explicites) sur IBM i

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_1 et est appelée par le gestionnaire de files d'attente pour extraire les droits dont dispose un groupe nommé pour accéder à un objet spécifié (mais sans les droits supplémentaires du groupe **personne**) ou les droits dont dispose le groupe principal du principal nommé pour accéder à un objet spécifié.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_GET\_EXPLICIT\_AUTHORITY.

### Syntaxe

**MQZ\_GET\_EXPLICIT\_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName, ObjectType, Authority, ComponentData, Continuation, CompCode, Reason)**

### Paramètres

L'appel MQZ\_GET\_EXPLICIT\_AUTHORITY comporte les paramètres suivants.

**QMgrName (MQCHAR48)-entrée**

Nom du gestionnaire de files d'attente

Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

**EntityName (MQCHAR12)-entrée**

Nom de l'entité.

Nom de l'entité à partir de laquelle l'accès à l'objet doit être extrait. La longueur maximale de la chaîne est de 12 caractères ; si elle est plus courte, elle est remplie à droite avec des blancs. Le nom ne se termine pas par un caractère NULL.

**EntityType (MQLONG)-entrée**

Type d'entité.

Type d'entité spécifié par *EntityName*. La valeur suivante peut être spécifiée:

**MQZAET\_PRINCIPAL**

Entité.

**GROUPE MQZAET\_GROUP**

.

**ObjectName (MQCHAR48)-entrée**

Nom de l'objet.

Nom de l'objet pour lequel les droits de l'entité doivent être extraits. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT\_Q\_MGR, ce nom est identique à *QMgrName*.

**ObjectType (MQLONG)-entrée**

Type d'objet.

Type d'entité spécifié par *ObjectName*. Il peut s'agir de :

**INFO MQOT\_AUTH\_INFO**

Informations d'authentification.

**CANAL\_MQTON**

Canal.

**MQOT\_CLNTCONN\_CHANNEL**

Canal de connexion client.

**MQOT\_PROGRAMME d'écoute**

Programme d'écoute.

**MQOT\_NAMELIST**

NAMELIST.

**PROCESSUS MQ**

Définition de processus.

**MQOT\_Q**

File d'attente.

**MQOT\_Q\_DIR**

Gestionnaire de files d'attente.

**SERVICE MQOT**

Service.

**Droits d'accès (MQLONG)-sortie**

Autorité de l'entité.

Si l'entité dispose d'un droit d'accès, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO\_\*). S'il possède plusieurs droits, cette zone correspond au OU bit à bit des constantes MQZAO\_\* correspondantes.

**ComponentData (MQBYTE x ComponentDataLongueur)-entrée / sortie**

Données de composant.

Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce



composant sont conservées et présentées lors du prochain appel de l'une des fonctions de ce composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

#### **Continuation (MQLONG)-sortie**

Indicateur de continuation défini par composant.

Les valeurs suivantes peuvent être spécifiées :

##### **MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_GET\_EXPLICIT\_AUTHORITY, cela a le même effet que MQZCI\_CONTINUE.

##### **MQZCI\_CONTINUER**

Passez au composant suivant.

##### **MQZCI\_ARRET**

Ne passez pas au composant suivant.

#### **CompCode (MQLONG)-sortie**

Code achèvement.

Il peut s'agir de :

##### **MQCC\_OK**

Achèvement réussi.

##### **MQCC\_FAILED**

Echec de l'appel.

#### **Cause (MQLONG)-sortie**

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

##### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

##### **MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') Non autorisé pour l'accès.

##### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

##### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

##### **MQRC\_UNKNOWN\_ENTITY**

(2292, X'8F4') Entité inconnue du service.

Pour plus d'informations sur ces codes anomalie, voir [Messages et codes anomalie](#).

## **Appel C**

```
MQZ_GET_EXPLICIT_AUTHORITY (QMgrName, EntityName, EntityType,  
                             ObjectName, ObjectType, &Authority,  
                             ComponentData, &Continuation,  
                             &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48 QMgrName;          /* Queue manager name */  
MQCHAR12 EntityName;       /* Entity name */  
MQLONG   EntityType;      /* Entity type */
```

```

MQCHAR48  ObjectName;      /* Object name */
MQLONG    ObjectType;     /* Object type */
MQLONG    Authority;      /* Authority of entity */
MQBYTE    ComponentData[n]; /* Component data */
MQLONG    Continuation;   /* Continuation indicator set by
                           component */
MQLONG    CompCode;       /* Completion code */
MQLONG    Reason;         /* Reason code qualifying CompCode */

```

## IBM i MQZ\_INIT\_AUTHORITY (Initialisation du service d'autorisation) sur IBM i

Cette fonction est fournie par un composant de service d'autorisation et est appelée par le gestionnaire de files d'attente lors de la configuration du composant. Il est prévu d'appeler MQZEP afin de fournir des informations au gestionnaire de files d'attente.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_INIT\_AUTHORITY.

### Syntaxe

**Commande MQZ\_INIT\_AUTHORITY** (*Hconfig, Options, QMgrName, ComponentDataLength, ComponentData, Version, CompCode, Reason*)

### Paramètres

L'appel MQZ\_INIT\_AUTHORITY comporte les paramètres suivants.

#### Hconfig (MQHCONFIG)-entrée

Descripteur de configuration.

Cette poignée représente le composant particulier en cours d'initialisation. Il doit être utilisé par le composant lors de l'appel du gestionnaire de files d'attente avec la fonction MQZEP.

#### Options (MQLONG)-entrée

Options d'initialisation.

Il peut s'agir de :

##### MQZIO\_PRIMAIRE

Initialisation principale.

##### MQZIO\_SECONDAIRE

Initialisation secondaire.

#### QMgrName (MQCHAR48)-entrée

Nom du gestionnaire de files d'attente

Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

#### ComponentDataLongueur (MQLONG)-entrée

Longueur des données de composant.

Longueur en octets de la zone *ComponentData* . Cette longueur est définie dans les données de configuration du composant.

#### ComponentData (MQBYTE x ComponentDataLongueur)-entrée / sortie

Données de composant.

Il est initialisé à tous les zéros avant d'appeler la fonction d'initialisation principale du composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions (y compris la

fonction d'initialisation) fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une des fonctions de ce composant.

### **Version (MQLONG)-entrée / sortie**

Numéro de version.

En entrée de la fonction d'initialisation, identifie le numéro de version *le plus élevé* pris en charge par le gestionnaire de files d'attente. La fonction d'initialisation doit modifier, si nécessaire, la version de l'interface prise en charge par *elle*. Si le gestionnaire de files d'attente ne prend pas en charge la version renvoyée par le composant, il appelle la fonction MQZ\_TERM\_AUTHORITY du composant et n'utilise plus ce composant.

Les valeurs suivantes sont prises en charge :

#### **MQZAS\_VERSION\_1**

Version 1.

#### **MQZAS\_VERSION\_2**

Version 2.

#### **MQZAS\_VERSION\_3**

Version 3.

#### **MQZAS\_VERSION\_4**

Version 4.

#### **MQZAS\_VERSION\_5**

Version 5.

#### **MQZAS\_VERSION\_6**

IBM WebSphere MQ 6.

### **CompCode (MQLONG)-sortie**

Code achèvement.

Il peut s'agir de :

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

### **Cause (MQLONG)-sortie**

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_INITIALIZATION\_FAILED**

(2286, X'8EE') L'initialisation a échoué pour une raison non définie.

#### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir [Messages et codes anomalie](#).

## **Appel C**

```
MQZ_INIT_AUTHORITY (Hconfig, Options, QMgrName, ComponentDataLength,  
                    ComponentData, &Version, &CompCode,  
                    &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```

MQHCONFIG  Hconfig;           /* Configuration handle */
MQLONG     Options;         /* Initialization options */
MQCHAR48   QMgrName;       /* Queue manager name */
MQLONG     ComponentDataLength; /* Length of component data */
MQBYTE     ComponentData[n]; /* Component data */
MQLONG     Version;        /* Version number */
MQLONG     CompCode;       /* Completion code */
MQLONG     Reason;         /* Reason code qualifying CompCode */

```

## IBM i MQZ\_INQUIRE (interrogation du service d'autorisation) sur IBM i

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_5 et est appelée par le gestionnaire de files d'attente pour demander la fonctionnalité prise en charge. Lorsque plusieurs composants de service sont utilisés, les composants de service sont appelés dans l'ordre inverse de l'ordre dans lequel ils ont été installés.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_INQUIRE.

### Syntaxe

#### MQZ\_INQUIRE

*(QMgrName, SelectorCount, Selectors, IntAttrCount, IntAttrs, CharAttrLength, CharAttrs, SelectorReturned, ComponentData, Continuation, CompCode, Reason)*

### Paramètres

L'appel MQZ\_INQUIRE comporte les paramètres suivants.

#### QMgrName (MQCHAR48)-entrée

Nom du gestionnaire de files d'attente

Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

#### SelectorCount (MQLONG)-entrée

Nombre de sélecteurs.

Nombre de sélecteurs fournis dans le paramètre Sélecteurs.

La valeur doit être comprise entre zéro et 256.

#### Sélecteurs (MQLONG x SelectorCount)-entrée

Sélecteurs.

Tableau de sélecteurs. Chaque sélecteur identifie un attribut requis et doit être de l'un des types suivants:

- MQIACF\_ \* (entier)
- MQCACF\_ \* (caractère)

Les sélecteurs peuvent être spécifiés dans n'importe quel ordre. Le nombre de sélecteurs dans le tableau est indiqué par le paramètre SelectorCount .

Les attributs de type entier identifiés par les sélecteurs sont renvoyés dans le paramètre IntAttrs dans l'ordre dans lequel ils apparaissent dans les sélecteurs.

Les attributs de caractères identifiés par les sélecteurs sont renvoyés dans le paramètre CharAttrs dans l'ordre dans lequel ils apparaissent dans les sélecteurs.

#### IntAttrCount (MQLONG)-entrée

Nombre d'attributs de type entier.

Nombre d'attributs de type entier fournis dans le paramètre `IntAttrs` .

La valeur doit être comprise entre 0 et 256.

#### **IntAttrs (Nombre IntAttrMQLONG x)-sortie**

Attributs de type entier.

Tableau d'attributs de type entier. Les attributs d'entier sont renvoyés dans le même ordre que les sélecteurs d'entier correspondants dans le tableau `Sélecteurs`.

#### **CharAttrCount (MQLONG)-entrée**

Longueur de la mémoire tampon des attributs de caractères.

Longueur en octets du paramètre `CharAttrs` .

La valeur doit au moins correspondre à la somme des longueurs des attributs de caractères demandés. Si aucun attribut de caractère n'est demandé, zéro est une valeur valide.

#### **CharAttrs (Nombre CharAttrMQLONG x)-sortie**

Mémoire tampon des attributs de caractères.

Mémoire tampon contenant des attributs de caractères, concaténés ensemble. Les attributs de caractères sont renvoyés dans le même ordre que les sélecteurs de caractères correspondants dans le tableau `Sélecteurs`.

La longueur de la mémoire tampon est indiquée par le paramètre `CharAttrCount`.

#### **SelectorReturned (MQLONGxSelectorCount)-entrée**

Sélecteur renvoyé.

Tableau de valeurs identifiant les attributs qui ont été renvoyés à partir de l'ensemble demandé par les sélecteurs dans le paramètre `Sélecteurs`. Le nombre de valeurs de ce tableau est indiqué par le paramètre `SelectorCount` . Chaque valeur du tableau est liée au sélecteur à partir de la position correspondante dans le tableau `Sélecteurs`. Chaque valeur est l'une des suivantes:

##### **MQZSL\_RENVOYE**

L'attribut demandé par le sélecteur correspondant dans le paramètre `Sélecteurs` a été renvoyé.

##### **MQZSL\_NOT\_RETURNED**

L'attribut demandé par le sélecteur correspondant dans le paramètre `Sélecteurs` n'a pas été renvoyé.

Le tableau est initialisé avec toutes les valeurs `MQZSL_NOT_RETURNED`. Lorsqu'un composant de service d'autorisation renvoie un attribut, il définit la valeur appropriée dans le tableau sur `MQZSL_RETURNED`. Cela permet à tout autre composant de service d'autorisation, auquel l'appel d'interrogation est envoyé, d'identifier les attributs qui ont déjà été renvoyés.

#### **ComponentData (MQBYTE x ComponentDataLongueur)-entrée / sortie**

Données de composant.

Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors du prochain appel de l'une des fonctions de ce composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre `ComponentDataLength` de l'appel `MQZ_INIT_AUTHORITY`.

#### **Continuation (MQLONG)-sortie**

Indicateur de continuation.

Les valeurs suivantes peuvent être spécifiées :

##### **MQZCI\_DEFAULT**

Continuation dépendante des autres composants.

##### **MQZCI\_ARRET**

Ne passez pas au composant suivant.

### **CompCode (MQLONG)-sortie**

Code achèvement.

Il peut s'agir de :

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_WARNING**

Achèvement partiel.

#### **MQCC\_FAILED**

Echec de l'appel.

### **Cause (MQLONG)-sortie**

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_WARNING:

#### **MQRC\_CHAR\_ATTRS\_TOO\_SHORT**

Espace insuffisant pour les attributs de caractères.

#### **MQRC\_INT\_COUNT\_TOO\_SMALL**

Espace insuffisant pour les attributs de type entier.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_SELECTOR\_COUNT\_ERREUR**

Le nombre de sélecteurs n'est pas valide.

#### **MQRC\_SELECTOR\_ERREUR**

Sélecteur d'attribut non valide.

#### **MQRC\_SELECTOR\_LIMIT\_EXCEEDED**

Trop de sélecteurs spécifiés.

#### **MQRC\_INT\_ATTR\_COUNT\_ERROR**

Le nombre d'attributs de type entier n'est pas valide.

#### **MQRC\_INT\_ATTRS\_ARRAY\_ERREUR**

Tableau d'attributs d'entier non valide.

#### **MQRC\_CHAR\_ATTR\_LENGTH\_ERROR**

Le nombre d'attributs de caractères n'est pas valide.

#### **MQRC\_CHAR\_ATTRS\_ERREUR**

La chaîne d'attributs de caractères n'est pas valide.

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

## **Appel C**

```
MQZ_INQUIRE (QMgrName, SelectorCount, Selectors, IntAttrCount,  
              &IntAttrs, CharAttrLength, &CharAttrs,  
              SelectorReturned, ComponentData, &Continuation,  
              &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48  QMgrName;           /* Queue manager name */  
MQLONG    SelectorCount;     /* Selector count */  
MQLONG    Selectors[n];      /* Selectors */  
MQLONG    IntAttrCount;      /* IntAttrs count */
```

```

MQLONG      IntAttrs[n];          /* Integer attributes */
MQLONG      CharAttrCount;       /* CharAttr count */
MQLONG      CharAttrs[n];       /* Character attributes */
MQLONG      SelectorReturned[n]; /* Selector returned */
MQBYTE      ComponentData[n];   /* Component data */
MQLONG      Continuation;       /* Continuation indicator set by
                                component */
MQLONG      CompCode;           /* Completion code */
MQLONG      Reason;            /* Reason code qualifying CompCode */

```

## **IBM i** **MQZ\_REFRESH\_CACHE (actualisation de toutes les autorisations)** sur IBM i

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_3 . Il est appelé par le gestionnaire de files d'attente pour actualiser la liste des autorisations détenues en interne par le composant.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_REFRESH\_CACHE (8L).

### Syntaxe

#### **MQZ\_CACHE d'actualisation**

*(QMgrName, ComponentData, Continuation, CompCode, Motif)*

### Paramètres

#### **QMgrName (MQCHAR48)-entrée**

Nom du gestionnaire de files d'attente

Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

#### **ComponentData (MQBYTE x ComponentDataLongueur ) -entrée / sortie**

Données de composant.

Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier. Toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées la prochaine fois qu'une fonction du composant est appelée.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre *ComponentDataLength* de l'appel MQZ\_INIT\_AUTHORITY.

#### **Continuation (MQLONG)-sortie**

Indicateur de continuation défini par composant.

Les valeurs suivantes peuvent être spécifiées :

##### **MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_REFRESH\_CACHE, cela a le même effet que MQZCI\_CONTINUE.

##### **MQZCI\_CONTINUER**

Passez au composant suivant.

##### **MQZCI\_ARRET**

Ne passez pas au composant suivant.

#### **CompCode (MQLONG)-sortie**

Code achèvement.

Il peut s'agir de :

**MQCC\_OK**

Achèvement réussi.

**MQCC\_FAILED**

Echec de l'appel.

**Cause (MQLONG)-sortie**

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

**MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

**MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

**Appel C**

```
MQZ_REFRESH_CACHE (QMgrName, ComponentData,
                  &Continuation, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQCHAR48  QMgrName;           /* Queue manager name */
MQBYTE    ComponentData[n]; /* Component data */
MQLONG    Continuation;      /* Continuation indicator set by
                             component */
MQLONG    CompCode;          /* Completion code */
MQLONG    Reason;           /* Reason code qualifying CompCode */
```

**IBM i MQZ\_SET\_AUTHORITY (Définition des droits) sur IBM i**

Cette fonction est fournie par un composant de service d'autorisation MQZAS\_VERSION\_1 et est appelée par le gestionnaire de files d'attente pour définir les droits d'accès d'une entité à l'objet spécifié.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_SET\_AUTHORITY.

**Remarque :** Cette fonction remplace les droits existants. Pour conserver les droits existants, vous devez les définir à nouveau à l'aide de cette fonction.

**Syntaxe**

**MQZ\_SET\_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName, ObjectType, Authority, ComponentData, Continuation, CompCode, Reason)**

**Paramètres**

L'appel MQZ\_SET\_AUTHORITY comporte les paramètres suivants.

**QMgrName (MQCHAR48)-entrée**

Nom du gestionnaire de files d'attente

Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

**EntityName (MQCHAR12)-entrée**

Nom de l'entité.



Nom de l'entité pour laquelle l'accès à l'objet doit être défini. La longueur maximale de la chaîne est de 12 caractères ; si elle est plus courte, elle est remplie à droite avec des blancs. Le nom ne se termine pas par un caractère NULL.

#### **EntityType (MQLONG)-entrée**

Type d'entité.

Type d'entité spécifié par *EntityName*. La valeur suivante peut être spécifiée:

##### **MQZAET\_PRINCIPAL**

Entité.

##### **GROUPE MQZAET\_GROUP**

.

#### **ObjectName (MQCHAR48)-entrée**

Nom de l'objet.

Nom de l'objet auquel l'accès est requis. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT\_Q\_MGR, ce nom est identique à *QMgrName*.

#### **ObjectType (MQLONG)-entrée**

Type d'objet.

Type d'entité spécifié par *ObjectName*. Il peut s'agir de :

##### **INFO MQOT\_AUTH\_INFO**

Informations d'authentification.

##### **CANAL\_MQTON**

Canal.

##### **MQOT\_CLNTCONN\_CHANNEL**

Canal de connexion client.

##### **MQOT\_PROGRAMME d'écoute**

Programme d'écoute.

##### **MQOT\_NAMELIST**

NAMELIST.

##### **PROCESSUS MQ**

Définition de processus.

##### **MQOT\_Q**

File d'attente.

##### **MQOT\_Q\_DIR**

Gestionnaire de files d'attente.

##### **SERVICE MQOT**

Service.

#### **Droits d'accès (MQLONG)-entrée**

Droits à vérifier.

Si une autorisation est définie, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO\_\*). Si plusieurs autorisations sont définies, il s'agit du OU bit à bit des constantes MQZAO\_\* correspondantes.

#### **ComponentData (MQBYTE x ComponentDataLongueur)-entrée / sortie**

Données de composant.

Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors du prochain appel de l'une des fonctions de ce composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

### **Continuation (MQLONG)-sortie**

Indicateur de continuation défini par composant.

Les valeurs suivantes peuvent être spécifiées :

#### **MQZCI\_DEFAULT**

Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ\_SET\_AUTHORITY, cela a le même effet que MQZCI\_STOP.

#### **MQZCI\_CONTINUER**

Passez au composant suivant.

#### **MQZCI\_ARRET**

Ne passez pas au composant suivant.

### **CompCode (MQLONG)-sortie**

Code achèvement.

Il peut s'agir de :

#### **MQCC\_OK**

Achèvement réussi.

#### **MQCC\_FAILED**

Echec de l'appel.

### **Cause (MQLONG)-sortie**

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

#### **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

#### **MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') Non autorisé pour l'accès.

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

#### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

#### **MQRC\_UNKNOWN\_ENTITY**

(2292, X'8F4') Entité inconnue du service.

## **Appel C**

```
MQZ_SET_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,  
                  ObjectType, Authority, ComponentData,  
                  &Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQCHAR48 QMgrName;          /* Queue manager name */  
MQCHAR12 EntityName;       /* Entity name */  
MQLONG   EntityType;       /* Entity type */  
MQCHAR48 ObjectName;       /* Object name */  
MQLONG   ObjectType;       /* Object type */  
MQLONG   Authority;        /* Authority to be checked */  
MQBYTE   ComponentData[n]; /* Component data */  
MQLONG   Continuation;     /* Continuation indicator set by  
                           component */
```

```
MQLONG   CompCode;           /* Completion code */
MQLONG   Reason;            /* Reason code qualifying CompCode */
```

## MQZ\_TERM\_AUTHORITY-Arrêt du service d'autorisation

Cette fonction est fournie par un composant de service d'autorisation et est appelée par le gestionnaire de files d'attente lorsqu'il n'a plus besoin des services de ce composant. La fonction doit effectuer tout nettoyage requis par le composant.

L'identificateur de cette fonction (pour MQZEP) est MQZID\_TERM\_AUTHORITY.

### Syntaxe

**MQZ\_TERM\_AUTHORITY** (*Hconfig, Options, QMgrName, ComponentData, CompCode, Reason*)

### Paramètres

L'appel MQZ\_TERM\_AUTHORITY comporte les paramètres suivants.

#### **Hconfig (MQHCONFIG)-entrée**

Descripteur de configuration.

Cette poignée représente le composant particulier en cours d'arrêt.

#### **Options (MQLONG)-entrée**

Options de fin.

Il peut s'agir de :

##### **MQZTO\_PRIMAIRE**

Arrêt principal.

##### **MQZTO\_SECONDAIRE**

Arrêt secondaire.

#### **QMgrName (MQCHAR48)-entrée**

Nom du gestionnaire de files d'attente

Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

#### **ComponentData (MQBYTE x ComponentDataLongueur)-entrée / sortie**

Données de composant.

Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors du prochain appel de l'une des fonctions de ce composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre **ComponentDataLength** de l'appel MQZ\_INIT\_AUTHORITY.

Lorsque l'appel MQZ\_TERM\_AUTHORITY est terminé, le gestionnaire de files d'attente supprime ces données.

#### **CompCode (MQLONG)-sortie**

Code achèvement.

Il peut s'agir de :

##### **MQCC\_OK**

Achèvement réussi.

## **MQCC\_FAILED**

Echec de l'appel.

### **Cause (MQLONG)-sortie**

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC\_OK:

## **MQRC\_NONE**

(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC\_FAILED:

## **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') Service sous-jacent non disponible.

## **Echec de la commande MQRC\_TERMINATION\_FAILED**

(2287, X'8FF') L'arrêt a échoué pour une raison non définie.

Pour plus d'informations sur ces codes anomalie, voir [Messages et codes anomalie](#).

## **Appel C**

```
MQZ_TERM_AUTHORITY (Hconfig, Options, QMgrName, ComponentData,  
                    &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
MQHCONFIG  Hconfig;           /* Configuration handle */  
MQLONG     Options;          /* Termination options */  
MQCHAR48   QMgrName;        /* Queue manager name */  
MQBYTE     ComponentData[n]; /* Component data */  
MQLONG     CompCode;        /* Completion code */  
MQLONG     Reason;          /* Reason code qualifying CompCode */
```

IBM i

## **MQZAC (contexte d'application) sur IBM i**

Ce paramètre spécifie les données relatives à l'application appelante.

La structure MQZAC est utilisée dans l'appel MQZ\_AUTHENTICATE\_USER pour le paramètre **ApplicationContext**.

### **Zones**

#### **StrucId (MQCHAR4)**

Identificateur de structure.

La valeur est:

#### **ID\_STRUC\_MQZAC**

Identificateur de la structure de contexte d'application.

Pour le langage de programmation C, la constante MQZAC\_STRUC\_ID\_ARRAY est également définie ; elle a la même valeur que MQZAC\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit d'une zone d'entrée pour le service.

#### **Version (MQLONG)**

Numéro de version de la structure.

La valeur est:

#### **MQZAC\_VERSION\_1**

Structure de contexte d'application Version-1 .

La constante suivante indique le numéro de version de la version en cours:

**MQZAC\_CURRENT\_VERSION**

Version actuelle de la structure de contexte d'application.

Il s'agit d'une zone d'entrée pour le service.

**ProcessId (MQPID)**

Identificateur de processus.

Identificateur de processus de l'application.

**ThreadId (MQTID)**

Identificateur d'unité d'exécution.

Identificateur d'unité d'exécution de l'application.

**ApplName (MQCHAR28)**

Nom de l'application.

Nom de l'application.

**UserID (MQCHAR12)**

ID utilisateur.

Pour les systèmes IBM i , le profil utilisateur sous lequel le travail d'application a été créé. (Sur IBM i, lorsqu'une permutation de profil est effectuée avec l'API QWTSETP dans le travail d'application, le profil utilisateur en cours est renvoyé).

**ID EffectiveUser(MQCHAR12)**

Identificateur utilisateur effectif.

Pour les systèmes IBM i , le profil utilisateur en cours du travail d'application.

**Environnement (MQLONG)**

Environnement.

Cette zone indique l'environnement à partir duquel l'appel a été effectué.

Cette valeur peut être l'une des suivantes:

**SERVEUR\_COMMAND\_MQ**

Serveur de commandes.

**MQXE\_MQSC**

Interpréteur de commandes runmqsc .

**MQXE\_MCA**

Agent MCA

**MQXE\_AUTRES**

Environnement non défini

**CallerType (MQLONG)**

Type d'appelant.

Cette zone indique le type de programme qui a effectué l'appel.

Cette valeur peut être l'une des suivantes:

**MQXACT\_EXTERNAL**

L'appel est externe au gestionnaire de files d'attente.

**MQXACT\_INTERNAL**

L'appel est interne au gestionnaire de files d'attente.

**AuthenticationType (MQLONG)**

Type d'authentification.

Cette zone indique le type d'authentification en cours d'exécution.

Cette valeur peut être l'une des suivantes:

### **MQZAT\_INITIAL\_CONTEXT**

L'appel d'authentification est dû à l'initialisation du contexte utilisateur. Cette valeur est utilisée lors d'un appel MQCONN ou MQCONNX.

### **MQZAT\_CHANGE\_CONTEXT**

L'appel d'authentification est dû à la modification du contexte de l'utilisateur. Cette valeur est utilisée lorsque l'agent MCA modifie le contexte utilisateur.

v

### **BindType (MQLONG)**

Type de liaison.

Cette zone indique le type de liaison en cours d'utilisation.

Cette valeur peut être l'une des suivantes:

#### **MQCNO\_FASTPATH\_BINDING**

Liaison Fastpath.

#### **MQCNO\_LIEN\_PARTAGE**

Liaison partagée.

#### **MQCNO\_LIEN\_ISOLÉ\_LIAISON**

Liaison isolée.

## **Déclaration C**

```
typedef struct tagMQZAC MQZAC;
struct tagMQZAC {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;          /* Structure version number */
    MQPID      ProcessId;        /* Process identifier */
    MQTID      ThreadId;         /* Thread identifier */
    MQCHAR28   ApplName;         /* Application name */
    MQCHAR12   UserID;           /* User identifier */
    MQCHAR12   EffectiveUserID;  /* Effective user identifier */
    MQLONG     Environment;      /* Environment */
    MQLONG     CallerType;       /* Caller type */
    MQLONG     AuthenticationType; /* Authentication type */
    MQLONG     BindType;         /* Bind type */
};
```

## **IBM i MQZAD (données de droits d'accès) sur IBM i**

La structure MQZAD est utilisée dans l'appel MQZ\_ENUMERATE\_AUTHORITY\_DATA pour deux paramètres.

Pour plus d'informations sur les paramètres **Filter** et **AuthorityBuffer**, voir «MQZ\_ENUMERATE\_AUTHORITY\_DATA (Enumération des données de droits d'accès) sur IBM i», à la page 1778 :

- MQZAD est utilisé pour le paramètre **Filter** qui est l'entrée de l'appel. Ce paramètre indique les critères de sélection à utiliser pour sélectionner les données de droits d'accès renvoyées par l'appel.
- MQZAD est également utilisé pour le paramètre **AuthorityBuffer** qui est généré à partir de l'appel. Ce paramètre indique les autorisations pour une combinaison de nom de profil, de type d'objet et d'entité.

## **Zones**

### **StrucId (MQCHAR4)**

Identificateur de structure.

La valeur est:

### **ID\_STRUC\_MQZ**

Identificateur de la structure de données de droits d'accès.

Pour le langage de programmation C, la constante MQZAD\_STRUC\_ID\_ARRAY est également définie ; elle a la même valeur que MQZAD\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit d'une zone d'entrée pour le service.

#### **Version (MQLONG)**

Numéro de version de la structure.

La valeur est:

##### **MQZAD\_VERSION\_1**

Structure de données de droits d'accès Version-1 .

La constante suivante indique le numéro de version de la version en cours:

##### **MQZAD\_CURRENT\_VERSION**

Version actuelle de la structure de données de droits d'accès.

Il s'agit d'une zone d'entrée pour le service.

#### **ProfileName (MQCHAR48)**

Nom du profil.

Pour le paramètre **Filter** , cette zone correspond au nom de profil à partir duquel les données de droits d'accès sont requises. Si le nom est entièrement à blanc jusqu'à la fin de la zone ou jusqu'au premier caractère NULL, les données de droits pour tous les noms de profil sont renvoyées.

Pour le paramètre **AuthorityBuffer** , cette zone correspond au nom d'un profil qui correspond aux critères de sélection spécifiés.

#### **ObjectType (MQLONG)**

Type d'objet.

Pour le paramètre **Filter** , cette zone correspond au type d'objet pour lequel des données de droits d'accès sont requises. Si la valeur est MQOT\_ALL, les données de droits d'accès pour tous les types d'objet sont renvoyées.

Pour le paramètre **AuthorityBuffer** , cette zone correspond au type d'objet auquel s'applique le profil identifié par **ProfileName** .

La valeur est l'une des suivantes ; pour le paramètre **Filter** , la valeur MQOT\_ALL est également valide:

##### **INFO MQOT\_AUTH\_INFO**

Informations d'authentification.

##### **CANAL\_MQTON**

Canal.

##### **MQOT\_CLNTCONN\_CHANNEL**

Canal de connexion client.

##### **MQOT\_PROGRAMME d'écoute**

Programme d'écoute.

##### **MQOT\_NAMELIST**

NAMELIST.

##### **PROCESSUS MQ**

Définition de processus.

##### **MQOT\_Q**

File d'attente.

##### **MQOT\_Q\_DIR**

Gestionnaire de files d'attente.

##### **SERVICE MQOT**

Service.

## Droits d'accès (MQLONG)

Droits d'accès.

Pour le paramètre **Filter**, cette zone est ignorée.

Pour le paramètre **AuthorityBuffer**, cette zone représente les autorisations dont dispose l'entité pour les objets identifiés par **ProfileName** et **ObjectType**. Si l'entité ne possède qu'un seul droit d'accès, la zone est égale à la valeur d'autorisation appropriée (constante MQZAO\_\*). Si l'entité possède plusieurs droits d'accès, la zone est le OU bit à bit des constantes MQZAO\_\* correspondantes.

## EntityDataPtr (PMQZED)

Adresse de la structure MQZED identifiant une entité.

Pour le paramètre **Filter**, cette zone pointe vers une structure MQZED qui identifie l'entité à partir de laquelle les données de droits d'accès sont requises. Si **EntityDataPtr** est le pointeur null, les données de droits d'accès de toutes les entités sont renvoyées.

Pour le paramètre **AuthorityBuffer**, cette zone pointe vers une structure MQZED qui identifie l'entité d'où proviennent les données de droits d'accès renvoyées.

## EntityType (MQLONG)

Type d'entité.

Pour le paramètre **Filter**, cette zone indique le type d'entité pour lequel des données de droits d'accès sont requises. Si la valeur est MQZAET\_NONE, les données de droits d'accès pour tous les types d'entité sont renvoyées.

Pour le paramètre **AuthorityBuffer**, cette zone indique le type de l'entité identifiée par la structure MQZED désignée par **EntityDataPtr**.

La valeur est l'une des suivantes ; pour le paramètre **Filter**, la valeur MQZAET\_NONE est également valide:

### MQZAET\_PRINCIPAL

Entité.

### GRUPE MQZAET\_GROUP

.

## Options (MQAUTHOPT)

Options.

Cette zone indique les options qui permettent de contrôler les profils affichés.

L'un des éléments suivants doit être spécifié:

### MQAUTHOPT\_NOM\_ALL\_MATCHING

Affiche tous les profils

### MQAUTHOPT\_NOM\_EXPLICITE

Affiche les profils ayant exactement le même nom que celui spécifié dans la zone **ProfileName**.

En outre, l'un des éléments suivants doit également être spécifié:

### MQAUTHOPT\_ENTITY\_SET

Affiche tous les profils utilisés pour calculer les droits cumulés de l'entité sur l'objet spécifié par **ProfileName**. La zone **ProfileName** ne doit pas contenir de caractères génériques.

- Si l'entité spécifiée est un principal, pour chaque membre de l'ensemble {entity, groups}, le profil le plus applicable qui s'applique à l'objet est affiché.
- Si l'entité spécifiée est un groupe, le profil le plus applicable du groupe qui s'applique à l'objet est affiché.
- Si cette valeur est spécifiée, les valeurs de **ProfileName**, **ObjectType**, **EntityType** et le nom d'entité spécifié dans la structure **EntityDataPtr** MQZED doivent toutes être non vides.

Si vous avez spécifié *MQAUTHOPT\_NAME\_ALL\_MATCHING*, vous pouvez également spécifier les éléments suivants:



## **MQAUTHOPT\_ENTITY\_EXPLICIT**

Affiche les profils ayant exactement le même nom d'entité que le nom d'entité spécifié dans la structure **EntityDataPtr** MQZED.

## **Déclaration C**

```
typedef struct tagMQZAD MQZAD;
struct tagMQZAD {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;          /* Structure version number */
    MQCHAR48   ProfileName;     /* Profile name */
    MQLONG     ObjectType;       /* Object type */
    MQLONG     Authority;        /* Authority */
    PMQZED     EntityDataPtr;    /* Address of MQZED structure identifying an
                                   entity */
    MQLONG     EntityType;       /* Entity type */
    MQAUTHOPT  Options;         /* Options */
};
```

## **IBM i MQZED (descripteur d'entité) sur IBM i**

La structure MQZED est utilisée dans un certain nombre d'appels de service d'autorisation pour spécifier l'entité pour laquelle l'autorisation doit être vérifiée.

### **Zones**

#### **StrucId (MQCHAR4)**

Identificateur de structure.

La valeur est:

#### **ID\_STRUC\_MQZ**

Identificateur de la structure de descripteur d'entité.

Pour le langage de programmation C, la constante MQZED\_STRUC\_ID\_ARRAY est également définie ; elle a la même valeur que MQZED\_STRUC\_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit d'une zone d'entrée pour le service.

#### **Version (MQLONG)**

Numéro de version de la structure.

La valeur est:

#### **MQZED\_VERSION\_1**

Structure de descripteur d'entité Version-1 .

La constante suivante indique le numéro de version de la version en cours:

#### **VERSION\_MQZED\_CURRENT\_VERSION**

Version actuelle de la structure de descripteur d'entité.

Il s'agit d'une zone d'entrée pour le service.

#### **EntityNamePtr (PMQCHAR)**

Adresse du nom de l'entité.

Il s'agit d'un pointeur vers le nom de l'entité dont l'autorisation doit être vérifiée.

#### **EntityDomainPtr (PMQCHAR)**

Adresse du nom de domaine de l'entité.

Il s'agit d'un pointeur vers le nom du domaine contenant la définition de l'entité dont l'autorisation doit être vérifiée.

### **SecurityId (MQBYTE40)**

Identificateur de sécurité.

Il s'agit de l'identificateur de sécurité dont l'autorisation doit être vérifiée.

### **CorrelationPtr (MQPTR)**

Pointeur de corrélation.

Cela facilite la transmission de données corrélationnelles entre la fonction d'authentification de l'utilisateur et d'autres fonctions OAM appropriées.

## **Déclaration C**

```
typedef struct tagMQZED MQZED;
struct tagMQZED {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;         /* Structure version number */
    PMQCHAR    EntityNamePtr;   /* Address of entity name */
    PMQCHAR    EntityDomainPtr; /* Address of entity domain name */
    MQBYTE40   SecurityId;     /* Security identifier */
    MQPTR      CorrelationPtr;  /* Address of correlation data */
}
```

## **IBM i MQZFP (Free parameters) sous IBM i**

Ce paramètre indique les données relatives à la ressource à libérer.

La structure MQZFP est utilisée dans l'appel MQZ\_FREE\_USER pour le paramètre **FreeParms** .

## **Zones**

### **StrucId (MQCHAR4)**

Identificateur de structure.

La valeur est:

#### **ID\_STRUC\_MQZFP**

Identificateur de la structure des paramètres libres.

Pour le langage de programmation C, la constante MQZFP\_STRUC\_ID\_ARRAY est également définie ; elle a la même valeur que MQZFP\_STRUC\_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

Il s'agit d'une zone d'entrée pour le service.

### **Version (MQLONG)**

Numéro de version de la structure.

La valeur est:

#### **MQZFP\_VERSION\_1**

Structure des paramètres libres Version-1 .

La constante suivante indique le numéro de version de la version en cours:

#### **VERSION\_MQZFP\_CURRENT\_VERSION**

Version actuelle de la structure des paramètres libres.

Il s'agit d'une zone d'entrée pour le service.

### **Réservé (MQBYTE8)**

Zone réservée.

La valeur initiale est null.

### **CorrelationPtr (MQPTR)**

Pointeur de corrélation.

Adresse des données de corrélation relatives à la ressource à libérer.

## Déclaration C

```
typedef struct tagMQZFP MQZFP;
struct tagMQZFP {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQBYTE8   Reserved;         /* Reserved field */
    MQPTR     CorrelationPtr;   /* Address of correlation data */
};
```

### MQZIC (contexte d'identité) sur IBM i

La structure MQZIC est utilisée dans l'appel MQZ\_AUTHENTICATE\_USER pour le paramètre **IdentityContext**.

La structure MQZIC contient des informations de contexte d'identité qui identifient l'utilisateur de l'application qui a inséré le message dans une file d'attente pour la première fois:

- Le gestionnaire de files d'attente remplit la zone UserIdentifier avec un nom qui identifie l'utilisateur, la manière dont le gestionnaire de files d'attente peut effectuer cette opération dépend de l'environnement dans lequel l'application s'exécute.
- Le gestionnaire de files d'attente remplit la zone AccountingToken avec un jeton ou un numéro qu'il a déterminé à partir de l'application qui a inséré le message.
- Les applications peuvent utiliser la zone de données ApplIdentity pour toute information supplémentaire qu'elles souhaitent inclure sur l'utilisateur (par exemple, un mot de passe chiffré).

Les applications dûment autorisées peuvent définir le contexte d'identité à l'aide de la fonction MQZ\_AUTHENTICATE\_USER.

Un identificateur de sécurité système Windows (SID) est stocké dans la zone AccountingToken lorsqu'un message est créé sous IBM MQ for Windows. Le SID peut être utilisé pour compléter la zone UserIdentifier et pour établir les données d'identification d'un utilisateur.

## Zones

### StrucId (MQCHAR4)

Identificateur de structure.

La valeur est:

#### **ID\_STRUC\_MQZ**

Identificateur de la structure de contexte d'identité.

Pour le langage de programmation C, la constante MQZIC\_STRUC\_ID\_ARRAY est également définie ; elle a la même valeur que MQZIC\_STRUC\_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

Il s'agit d'une zone d'entrée pour le service.

### Version (MQLONG)

Numéro de version de la structure.

La valeur est:

#### **MQZIC\_VERSION\_1**

Structure de contexte d'identité Version-1.

La constante suivante indique le numéro de version de la version en cours:

#### **MQZIC\_CURRENT\_VERSION**

Version actuelle de la structure de contexte d'identité.

Il s'agit d'une zone d'entrée pour le service.

### UserIdentifier (MQCHAR12)

ID utilisateur.

Il fait partie du **contexte d'identité** du message.

*UserIdentifier* indique l'ID utilisateur de l'application à l'origine du message. Le gestionnaire de files d'attente traite ces informations comme des données de type caractères, mais ne définit pas leur format. Pour plus d'informations sur la zone *UserIdentifier*, voir «[UserIdentifier \(MQCHAR12\) pour MQMD](#)», à la page 477.

### AccountingToken (MQBYTE32)

Jeton de comptabilité.

Il fait partie du **contexte d'identité** du message.

*AccountingToken* permet à une application de faire en sorte que le travail effectué à la suite du message soit correctement facturé. Le gestionnaire de files d'attente traite ces informations comme une chaîne de bits et ne vérifie pas leur contenu. Pour plus d'informations sur la zone *AccountingToken*, voir «[AccountingToken \(MQBYTE32\) pour MQMD](#)», à la page 478.

### Données ApplIdentity(MQCHAR32)

Données d'application relatives à l'identité.

Il fait partie du **contexte d'identité** du message.

*ApplIdentityData* est une information définie par la suite d'applications qui peut être utilisée pour fournir des informations supplémentaires sur l'origine du message. Par exemple, elle peut être définie par des applications s'exécutant avec des droits utilisateur appropriés pour indiquer si les données d'identité sont dignes de confiance. Pour plus d'informations sur la zone *ApplIdentityData*, voir «[Données ApplIdentity\(MQCHAR32\) pour MQMD](#)», à la page 480.

## Déclaration C

```
typedef struct tagMQZED MQZED;
struct tagMQZED {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQCHAR12  UserIdentifier;    /* User identifier */
    MQBYTE32  AccountingToken;  /* Accounting token */
    MQCHAR32  ApplIdentityData; /* Application data relating to identity */
};
```

## Les classes et interfaces IBM MQ .NET

Les classes et les interfaces IBM MQ .NET sont répertoriées par ordre alphabétique. Les propriétés, les méthodes et les constructeurs sont décrits.

### Classe MQAsyncStatus.NET

Utilisez *MQAsyncStatus* pour demander le statut de l'activité MQI précédente ; par exemple, demander la réussite des opérations d'insertion asynchrone précédentes. *MQAsyncStatus* encapsule les fonctions de la structure de données MQSTS .

#### Classe

```
System.Object
├── IBM.WMQ.MQBase
│   └── IBM.WMQ.MQBaseObject
│       └── IBM.WMQ.MQAsyncStatus
```

```
public class IBM.WMQ.MQAsyncStatus extends IBM.WMQ.MQBaseObject;
```

- «Propriétés», à la page 1805
- «Constructeurs», à la page 1805

## Propriétés

Test de l'émission de `MQException` lors de l'obtention des propriétés.

**public static int CompCode {get;}**

Code achèvement de la première erreur ou avertissement.

**public static int Reason {get;}**

Code raison de la première erreur ou avertissement.

**public static int PutSuccessCount {get;}**

Nombre d'appels MQI asynchrones réussis.

**public static int PutWarningCount {get;}**

Nombre d'appels d'insertion MQI asynchrones ayant abouti avec un avertissement.

**public static int PutFailureCount {get;}**

Nombre d'appels MQI asynchrones ayant échoué.

**public static int ObjectType {get;}**

Type d'objet de la première erreur. Les valeurs suivantes sont possibles :

- `MQC.MQOT_ALIAS_Q`
- `MQC.MQOT_LOCAL_Q`
- `MQC.MQOT_MODEL_Q`
- `MQC.MQOT_Q`
- `MQC.MQOT_REMOTE_Q`
- `MQC.MQOT_TOPIC`
- 0, ce qui signifie qu'aucun objet n'est renvoyé

**public static string ObjectName {get;}**

Nom de l'objet.

**public static string ObjectQMgrName {get;}**

Nom du gestionnaire de files d'attente d'objets.

**public static string ResolvedObjectName {get;}**

Nom de l'objet résolu.

**public static string ResolvedObjectQMgrName {get;}**

Nom du gestionnaire de files d'attente d'objet résolu.

## Constructeurs

**public MQAsyncStatus() throws MQException;**

Méthode de construction: construit un objet avec des zones initialisées à zéro ou à blanc, selon le cas.

## Classe `MQAuthenticationInformationRecord.NET`

Utilisez `MQAuthenticationInformationRecord` pour spécifier des informations sur un authentificateur à utiliser dans une connexion client TLS IBM MQ .  
`MQAuthenticationInformationRecord` encapsule un enregistrement d'informations d'authentification, `MQAIR`.

## Classe

```
System.Object
└─ IBM.WMQ.MQAuthenticationInformationRecord
```

```
public class IBM.WMQ.MQAuthenticationInformationRecord extends System.Object;
```

- «Propriétés», à la page [1806](#)
- «Constructeurs», à la page [1806](#)

## Propriétés

Test de l'émission de `MQException` lors de l'obtention des propriétés.

```
public long Version {get; set;}
```

Numéro de version de la structure.

```
public long AuthInfoType {get; set;}
```

Type des informations d'authentification. Cet attribut doit être défini sur l'une des valeurs suivantes:

- OCSP -La vérification du statut de révocation de certificat est effectuée à l'aide d'OCSP.
- CRLLDAP -La vérification du statut de révocation de certificat est effectuée à l'aide des listes de révocation de certificat sur les serveurs LDAP.

```
public string AuthInfoConnName {get; set;}
```

Nom DNS ou adresse IP de l'hôte sur lequel le serveur LDAP s'exécute, avec un numéro de port facultatif. Ce mot clé est obligatoire.

```
public string LDAPPassword {get; set;}
```

Mot de passe associé au nom distinctif de l'utilisateur qui accède au serveur LDAP. Cette propriété s'applique uniquement lorsque `AuthInfoType` est défini sur CRLLDAP.

```
public string LDAPUserName {get; set;}
```

Nom distinctif de l'utilisateur qui accède au serveur LDAP. Lorsque vous définissez cette propriété, `LDAPUserNameLength` et `LDAPUserNamePtr` sont automatiquement définis correctement. Cette propriété s'applique uniquement lorsque `AuthInfoType` est défini sur CRLLDAP.

```
public string OCSPResponderURL {get; set;}
```

Adresse URL à laquelle le canal répondeur OCSP peut être contacté. Cette propriété s'applique uniquement lorsque `AuthInfoType` est défini sur OCSP

Cette zone est sensible à la casse. Il doit commencer par la chaîne `http://` en minuscules. Le reste de l'URL peut être sensible à la casse, en fonction de l'implémentation du serveur OCSP.

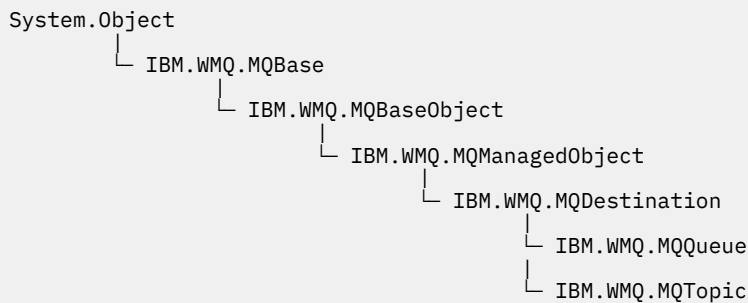
## Constructeurs

```
MQAuthenticationInformationRecord();
```

## Classe MQDestination.NET

Utilisez `MQDestination` pour accéder aux méthodes communes à `MQQueue` et `MQTopic`. `MQDestination` est une classe de base abstraite et ne peut pas être instanciée.

## Classe



```
public class IBM.WMQ.MQDestination extends IBM.WMQ.MQManagedObject;
```

- «Propriétés», à la page 1807
- «Méthodes», à la page 1807
- «Constructeurs», à la page 1809

## Propriétés

Test de l'émission de `MQException` lors de l'obtention des propriétés.

```
public DateTime CreationDateTime {get;}
```

Date et heure de création de la file d'attente ou de la rubrique. Initialement contenue dans `MQQueue`, cette propriété a été déplacée dans la classe `MQDestination` de base.

Il n'existe pas de valeur par défaut.

```
public int DestinationType {get;}
```

Valeur entière décrivant le type de destination utilisée. Initialisée à partir du constructeur de sous-classes, `MQQueue` ou `MQTopic`, cette valeur peut prendre l'une des valeurs suivantes:

- `MQOT_Q`
- `MQOT_TOPIC`

Il n'existe pas de valeur par défaut.

## Méthodes

```
public void Get(MQMessage message);
public void Get(MQMessage message, MQGetMessageOptions getMessageOptions);
public void Get(MQMessage message, MQGetMessageOptions getMessageOptions, int MaxMsgSize);
```

Emet `MQException`.

Extrait un message d'une file d'attente si la destination est un objet `MQQueue` ou d'une rubrique si la destination est un objet `MQTopic`, en utilisant une instance par défaut de `MQGetMessageOptions` pour effectuer l'extraction.

Si l'extraction échoue, l'objet `MQMessage` n'est pas modifié. Si elle aboutit, le descripteur de message et les parties de données de message du `MQMessage` sont remplacés par le descripteur de message et les données de message du message entrant.

Tous les appels à IBM MQ à partir d'un `MQQueueManager` particulier sont synchrones. Par conséquent, si vous effectuez une opération `get` avec attente, toutes les autres unités d'exécution utilisant le même `MQQueueManager` ne peuvent pas effectuer d'autres appels IBM MQ jusqu'à ce que l'appel `Get` soit effectué. Si vous avez besoin de plusieurs unités d'exécution pour accéder à IBM MQ simultanément, chaque unité d'exécution doit créer son propre objet `MQQueueManager`.

## Message

Contient le descripteur de message et les données de message renvoyées. Certaines des zones du descripteur de message sont des paramètres d'entrée. Il est important de s'assurer que les paramètres d'entrée `MessageId` et `CorrelationId` sont définis comme requis.

Un client reconnectable renvoie le code anomalie `MQRC_BACKED_OUT` après une reconnexion réussie, pour les messages reçus sous `MQGM_SYNCPOINT`.

## Options de `getMessage`

Options contrôlant l'action de la commande `get`.

L'utilisation de l'option `MQC.MQGMO_CONVERT` peut générer une exception avec le code anomalie `MQC.MQRC_CONVERTED_STRING_TOO_BIG` lors de la conversion de codes de caractères codés sur un octet en codes codés sur deux octets. Dans ce cas, le message est copié dans la mémoire tampon sans conversion.

Si `getMessageOptions` n'est pas spécifié, l'option de message utilisée est `MQGMO_NOWAIT`.

Si vous utilisez l'option `MQGMO_LOGICAL_ORDER` dans un client reconnectable, le code anomalie `MQRC_RECONNECT_INCOMPATIBLE` est renvoyé.

## Taille `MaxMsg`

Message le plus volumineux que cet objet message doit recevoir. Si le message de la file d'attente est supérieur à cette taille, l'un des deux événements suivants se produit:

- Si l'indicateur `MQGMO_ACCEPT_TRUNCATED_MSG` est défini dans l'objet `MQGetMessageOptions`, le message est rempli avec autant de données de message que possible. Une exception est émise avec le code achèvement `MQCC_WARNING` et le code anomalie `MQRC_TRUNCATED_MSG_ACCEPTED`.
- Si l'indicateur `MQGMO_ACCEPT_TRUNCATED_MSG` n'est pas défini, le message reste dans la file d'attente. Une exception est émise avec le code achèvement `MQCC_WARNING` et le code anomalie `MQRC_TRUNCATED_MSG_FAILED`.

Si `MaxMsgSize` n'est pas spécifié, l'intégralité du message est extraite.

```
public void Put(MQMessage message);  
public void Put(MQMessage message, MQPutMessageOptions putMessageOptions);
```

Emet `MQException`.

Insère un message dans une file d'attente si la destination est un objet `MQQueue` ou publie un message dans une rubrique si la destination est un objet `MQTopic`.

Les modifications apportées à l'objet `MQMessage` après l'appel d'insertion n'affectent pas le message réel dans la file d'attente IBM MQ ou dans la rubrique de publication.

`Put` met à jour les propriétés `MessageId` et `CorrelationId` de l'objet `MQMessage` et n'efface pas les données de message. Les appels `Put` ou `Get` supplémentaires font référence aux informations mises à jour dans l'objet `MQMessage`. Par exemple, dans le fragment de code suivant, le premier message contient `a` et le second `ab`.

```
msg.WriteString("a");  
q.Put(msg, pmo);  
msg.WriteString("b");  
q.Put(msg, pmo);
```

## Message

Objet `MQMessage` contenant les données du descripteur de message et le message à envoyer. Le descripteur de message peut être modifié à la suite de cette méthode. Les valeurs du descripteur de message immédiatement après la fin de cette méthode sont les valeurs qui ont été placées dans la file d'attente ou publiées dans la rubrique.

Les codes anomalie suivants sont renvoyés à un client reconnectable:

- `MQRC_CALL_INTERRUPTED` si la connexion est interrompue lors de l'exécution d'un appel `Put` sur un message persistant et que la reconnexion a abouti.



- MQRC\_NONE si la connexion aboutit lors de l'exécution d'un appel Put sur un message non persistant (voir [Application Recovery](#)).

### Options de putMessage

Options contrôlant l'action de l'insertion.

Si *putMessageOptions* n'est pas spécifié, l'instance par défaut de MQPutMessageOptions est utilisée.

Si vous utilisez l'option MQPMO\_LOGICAL\_ORDER dans un client reconnectable, le code anomalie MQRC\_RECONNECT\_INCOMPATIBLE est renvoyé.

**Remarque :** Pour plus de simplicité et de performances, si vous souhaitez placer un seul message dans une file d'attente, utilisez l'objet MQQueueManager.Put. Vous devez disposer d'un objet MQQueue pour cela.

## Constructeurs

MQDestination est une classe de base abstraite et ne peut pas être instanciée. Accédez à des destinations à l'aide de constructeurs MQQueue et MQTopic ou à l'aide de MQQueueManager.AccessQueue et MQQueueManager.AccessTopic methods.

## Classe MQEnvironment.NET

Utilisez MQEnvironment pour contrôler la façon dont le constructeur MQQueueManager est appelé et pour sélectionner une connexion IBM MQ MQI client. La classe MQEnvironment contient des propriétés qui contrôlent le comportement de IBM MQ.

### Classe

```
System.Object
└─ IBM.WMQ.MQEnvironment
```

```
public class IBM.WMQ.MQEnvironment extends System.Object;
```

- «Propriétés-client uniquement», à la page 1809
- «Propriétés», à la page 1810
- «Constructeurs», à la page 1811

### Propriétés-client uniquement

Test de l'émission de MQException lors de l'obtention des propriétés.

```
public static int CertificateValPolicy {get; set;}
```

Définissez les règles de validation de certificat TLS qui sont utilisées pour valider les certificats numériques reçus des systèmes partenaires distants. Les valeurs valides sont :

- MQC.CERTIFICATE\_VALIDATION\_POLICY\_ANY
- MQC.CERTIFICATE\_VALIDATION\_POLICY\_RFC5280

```
public static ArrayList EncryptionPolicySuiteB {get; set;}
```

Définissez le niveau de la cryptographie compatible Suite B. Les valeurs valides sont :

- MQC.MQ\_SUITE\_B\_NONE -Il s'agit de la valeur par défaut.
- MQC.MQ\_SUITE\_B\_128\_BIT
- MQC.MQ\_SUITE\_B\_192\_BIT

**public static string Channel {get; set;}**

Nom du canal pour la connexion au gestionnaire de files d'attente cible. Vous devez définir la propriété de canal avant d'instancier une instance `MQQueueManager` en mode client.

**public static int FipsRequired {get; set;}**

Spécifiez `MQC.MQSSL_FIPS_YES` pour utiliser uniquement les algorithmes certifiés FIPS si la cryptographie est effectuée dans IBM MQ. La valeur par défaut est `MQC.MQSSL_FIPS_NO`.

Si du matériel de cryptographie est configuré, les modules de cryptographie utilisés sont ceux fournis par le produit matériel. Selon le matériel utilisé, il se peut qu'ils ne soient pas certifiés FIPS à un niveau particulier.

**public static string Hostname {get; set;}**

Nom d'hôte TCP/IP de l'ordinateur sur lequel réside le serveur IBM MQ. Si le nom d'hôte n'est pas défini et qu'aucune propriété de remplacement n'est définie, le mode de liaison du serveur est utilisé pour la connexion au gestionnaire de files d'attente local.

**public static int Port {get; set;}**

Port auquel se connecter. Il s'agit du port sur lequel le serveur IBM MQ écoute les demandes de connexion entrantes. La valeur par défaut est 1414.

**public static string SSLCipherSpec {get; set;}**

Définissez `SSLCipherSpec` sur la valeur de `CipherSpec` définie sur le canal `SVRCONN` pour activer TLS pour la connexion. La valeur par défaut est `Null` et TLS n'est pas activé pour la connexion.

**public static string sslPeerName {get; set;}**

Modèle de nom distinctif. Si `sslCipherSpec` est défini, cette variable peut être utilisée pour s'assurer que le gestionnaire de files d'attente approprié est utilisé. Si la valeur est `null` (valeur par défaut), le nom distinctif du gestionnaire de files d'attente n'est pas exécuté. `sslPeerName` est ignoré si `sslCipherSpec` est `null`.

**V 9.3.0** **public static string SSLKeyRepository {get; set;}**

Texte en clair ou phrase passe chiffrée permettant d'accéder au référentiel de clés. Les phrases passe de référentiel de clés sont chiffrées pour être utilisées par les applications client à l'aide de l'utilitaire `runmqicred`.

Si `SSLKeyRepositoryPassword` est défini sur `null` (valeur par défaut), la valeur de la variable d'environnement `MQKEYRPWD` ou de l'attribut `SSLKeyRepositoryPassword` dans le fichier de configuration du client est utilisée.

**V 9.3.0** **public static string InitialKey {get; set;}**

Clé initiale utilisée pour chiffrer la phrase passe du référentiel de clés spécifiée dans `SSLKeyRepositoryPassword`.

La clé initiale doit être spécifiée si un fichier de clés initial a été spécifié lorsque la phrase passe du référentiel de clés a été chiffrée à l'aide de l'utilitaire `runmqicred`.

## Propriétés

Test de l'émission de `MQException` lors de l'obtention des propriétés.

**public static ArrayList HdrCompList {get; set;}**

Liste de compression de données d'en-tête

**public static int KeyResetCount {get; set;}**

Indique le nombre d'octets non chiffrés envoyés et reçus dans une conversation TLS avant que la clé secrète ne soit renégociée.

**public static ArrayList MQAIRArray {get; set;}**

Un tableau d'objets `MQAuthenticationInformationRecord`.

**public static ArrayList MsgCompList {get; set;}**

Liste de compression de données de message

**public static string Password {get; set;}**

Mot de passe à authentifier. Le mot de passe référencé à partir de la structure MQCSP est renseigné en définissant cette propriété de mot de passe.

**public static string ReceiveExit {get; set;}**

Un exit de réception permet d'examiner et de modifier les données reçues d'un gestionnaire de files d'attente. Il est généralement utilisé avec un exit d'émission correspondant au niveau du gestionnaire de files d'attente. Si ReceiveExit est défini sur null, aucun exit de réception n'est appelé.

**public static string ReceiveUserData {get; set;}**

Données utilisateur associées à un exit de réception. Limité à 32 caractères.

**public static string SecurityExit {get; set;}**

Un exit de sécurité permet de personnaliser les flux de sécurité qui se produisent lorsqu'une tentative de connexion à un gestionnaire de files d'attente est effectuée. Si SecurityExit est défini sur null, aucun exit de sécurité n'est appelé.

**public static string SecurityUserData {get; set;}**

Données utilisateur associées à un exit de sécurité. Limité à 32 caractères.

**public static string SendExit {get; set;}**

Un exit d'émission vous permet d'examiner ou de modifier les données envoyées à un gestionnaire de files d'attente. Il est généralement utilisé avec un exit de réception correspondant au niveau du gestionnaire de files d'attente. Si SendExit est défini sur null, aucun exit d'émission n'est appelé.

**public static string SendUserData {get; set;}**

Données utilisateur associées à un exit d'émission. Limité à 32 caractères.

**public static string SharingConversations {get; set;}**

La zone SharingConversations est utilisée sur les connexions à partir d'applications .NET , lorsque ces applications n'utilisent pas de table de définition de canal du client (CCDT).

SharingConversations détermine le nombre maximal de conversations pouvant être partagées sur un socket associé à cette connexion.

La valeur 0 signifie que le canal fonctionne comme avant IBM WebSphere MQ 7.0, en ce qui concerne le partage de conversation, la lecture anticipée et le signal de présence.

La zone est transmise dans la table de hachage des propriétés en tant que SHARING\_CONVERSATIONS\_PROPERTY, lors de l'instanciation d'un gestionnaire de files d'attente IBM MQ .

Si vous ne spécifiez pas SharingConversations, la valeur par défaut 10 est utilisée.

**public static string SSLCryptoHardware {get; set;}**

Définit le nom de la chaîne de paramètres requise pour configurer le matériel cryptographique présent sur le système. SSLCryptoHardware est ignoré si sslCipherSpec a la valeur null.

**public static string SSLKeyRepository {get; set;}**

Définissez le nom de fichier qualifié complet du référentiel de clés.

 Si l'extension de fichier n'est pas fournie, elle est supposée être .kdb

Si SSLKeyRepository est défini sur null (valeur par défaut), la variable d'environnement MQSSLKEYR du certificat est utilisée pour localiser le référentiel de clés.

**public static string UserId {get; set;}**

ID utilisateur à authentifier. L'ID utilisateur référencé à partir de la structure MQCSP est renseigné en définissant UserId. Authentifiez UserId à l'aide d'une API ou d'un exit de sécurité.

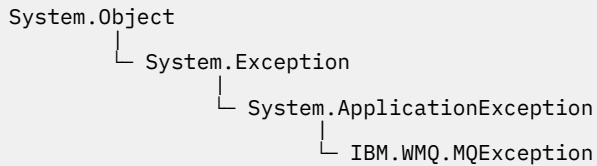
## Constructeurs

**public MQEnvironment()**

## Classe MQException.NET

Utilisez MQException pour rechercher le code achèvement et le code anomalie d'une fonction IBM MQ ayant échoué. Une exception MQException est émise chaque fois qu'une erreur IBM MQ se produit.

### Classe



```
public class IBM.WMQ.MQException extends System.ApplicationException;
```

- [«Propriétés», à la page 1812](#)
- [«Constructeurs», à la page 1812](#)

### Propriétés

```
public int CompletionCode {get; set;}
```

Code achèvement IBM MQ associé à l'erreur. Les valeurs possibles sont les suivantes:

- MQException.MQCC\_OK
- MQException.MQCC\_WARNING
- MQException.MQCC\_FAILED

```
public int ReasonCode {get; set;}
```

Code anomalie IBM MQ décrivant l'erreur.

### Constructeurs

```
public MQException(int completionCode, int reasonCode)
```

**completionCode**

Code achèvement IBM MQ .

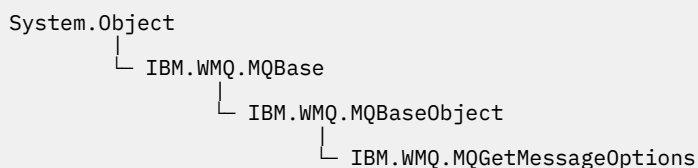
**reasonCode**

Code achèvement IBM MQ .

## Classe MQGetMessageOptions.NET

Utilisez MQGetMessageOptions pour spécifier le mode d'extraction des messages. Il modifie le comportement de MQDestination.Get.

### Classe



```
public class IBM.WMQ.MQGetMessageOptions extends IBM.WMQ.MQBaseObject;
```

- [«Propriétés», à la page 1813](#)
- [«Constructeurs», à la page 1815](#)

## Propriétés

**Remarque :** Le comportement de certaines des options disponibles dans cette classe dépend de l'environnement dans lequel elles sont utilisées. Ces éléments sont marqués d'un astérisque \*.

Test de l'émission de MQException lors de l'obtention des propriétés.

### **public int GroupStatus {get;}\***

GroupStatus indique si le message extrait se trouve dans un groupe et s'il s'agit du dernier du groupe. Les valeurs possibles sont les suivantes :

#### **MQC.MQGS\_LAST\_MSG\_IN\_GROUP**

Le message est le dernier ou le seul message du groupe.

#### **MQC.MQGS\_MSG\_IN\_GROUP**

Le message se trouve dans un groupe, mais n'est pas le dernier du groupe.

#### **MQC.MQGS\_NOT\_IN\_GROUP**

Le message n'est pas dans un groupe.

### **public int MatchOptions {get; set;}\***

MatchOptions détermine comment un message est sélectionné. Les options de correspondance suivantes peuvent être définies:

#### **MQC.MQMO\_MATCH\_CORREL\_ID**

ID de corrélation à mettre en correspondance.

#### **MQC.MQMO\_MATCH\_GROUP\_ID**

ID de groupe à mettre en correspondance.

#### **MQC.MQMO\_MATCH\_MSG\_ID**

ID message à mettre en correspondance.

#### **MQC.MQMO\_MATCH\_MSG\_SEQ\_NUMBER**

Correspond au numéro de séquence de message.

#### **MQC.MQMO\_NONE**

Aucune correspondance n'est requise.

### **public int Options {get; set;}**

Les options contrôlent l'action de MQQueue.get. Vous pouvez indiquer l'une des valeurs suivantes. Si plusieurs options sont requises, les valeurs peuvent être ajoutées ou combinées à l'aide de l'opérateur OR bit à bit.

#### **MQC.MQGMO\_ACCEPT\_TRUNCATED\_MSG**

Autoriser la troncature des données de message.

#### **MQC.MQGMO\_ALL\_MSGS\_AVAILABLE\***

Extraire des messages d'un groupe uniquement lorsque tous les messages du groupe sont disponibles.

#### **MQC.MQGMO\_ALL\_SEGMENTS\_AVAILABLE\***

Extraire les segments d'un message logique uniquement lorsque tous les segments du groupe sont disponibles.

#### **MQC.MQGMO\_BROWSE\_FIRST**

Parcourez depuis le début de la file d'attente.

#### **MQC.MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR\***

Parcourir le message sous le curseur de navigation.

#### **MQC.MQGMO\_BROWSE\_NEXT**

Parcourez la position en cours dans la file d'attente.

#### **MQC.MQGMO\_COMPLETE\_MSG\***

Extraire uniquement les messages logiques complets.

#### **MQC.MQGMO\_CONVERT**

Demandez la conversion des données d'application pour qu'elles soient conformes aux attributs CharSet et Codage de MQMessage, avant que les données ne soient copiées dans

la mémoire tampon de messages. Etant donné que la conversion de données est également appliquée lorsque les données sont extraites de la mémoire tampon de messages, les applications ne définissent pas cette option.

L'utilisation de cette option peut entraîner des problèmes lors de la conversion de jeux de caractères codés sur un octet en jeux de caractères codés sur deux octets. A la place, effectuez la conversion à l'aide des méthodes `readString`, `readLine` et `writeString` une fois que le message a été distribué.

**MQC.MQGMO\_FAIL\_IF QUIESCING**

Echec si le gestionnaire de files d'attente est au repos.

**MQC.MQGMO\_LOCK\***

Verrouillez le message parcouru.

**MQC.MQGMO\_LOGICAL\_ORDER\***

Renvoie des messages dans des groupes et des segments de messages logiques, dans l'ordre logique.

Si vous utilisez l'option `MQGMO_LOGICAL_ORDER` dans un client reconnectable, le code anomalie `MQRC_RECONNECT_INCOMPATIBLE` est renvoyé à l'application.

**MQC.MQGMO\_MARK\_SKIP\_BACKOUT\***

Permet l'annulation d'une unité de travail sans réinstanciation du message dans la file d'attente.

**MQC.MQGMO\_MSG\_UNDER\_CURSOR**

Obtenir le message sous le curseur de navigation.

**MQC.MQGMO\_NONE**

Aucune autre option n'a été spécifiée ; toutes les options prennent leurs valeurs par défaut.

**MQC.MQGMO\_NO\_PROPERTIES**

Aucune propriété du message, à l'exception des propriétés contenues dans le descripteur de message (ou l'extension), n'est extraite.

**MQC.MQGMO\_NO\_SYNCPOINT**

Obtenir un message sans contrôle de point de synchronisation.

**MQC.MQGMO\_NO\_WAIT**

Revenez immédiatement s'il n'y a pas de message approprié.

**MQC.MQGMO\_PROPERTIES\_AS\_Q\_DEF**

Extrayez les propriétés de message définies par l'attribut `PropertyControl` de `MQQueue`. L'accès aux propriétés de message dans le descripteur de message ou l'extension n'est pas affecté par l'attribut `PropertyControl`.

**MQC.MQGMO\_PROPERTIES\_COMPATIBILITY**

Extrayez les propriétés de message avec le préfixe `mcd`, `jms`, `usjro` ou `mqext`, dans les en-têtes `MQRFH2`. Les autres propriétés du message, à l'exception des propriétés contenues dans le descripteur de message ou l'extension, sont supprimées.

**MQC.MQGMO\_PROPERTIES\_FORCE\_MQRFH2**

Extrayez les propriétés de message, à l'exception des propriétés contenues dans le descripteur de message ou l'extension, dans les en-têtes `MQRFH2`. Utilisez `MQC.MQGMO_PROPERTIES_FORCE_MQRFH2` dans les applications qui s'attendent à extraire des propriétés mais qui ne peuvent pas être modifiées pour utiliser des descripteurs de message.

**MQC.MQGMO\_PROPERTIES\_IN\_HANDLE**

Extrayez les propriétés de message à l'aide d'un `MsgHandle`.

**MQC.MQGMO\_SYNCPOINT**

Obtenez le message sous contrôle de point de synchronisation. Le message est marqué comme étant indisponible pour d'autres applications, mais il est supprimé de la file d'attente uniquement lorsque l'unité de travail est validée. Le message est de nouveau disponible si l'unité d'oeuvre est annulée.

**MQC.MQGMO\_SYNCPOINT\_IF\_PERSISTENT\***

Obtenir le message avec le contrôle de point de synchronisation si le message est persistant.

**MQC.MQGMO\_UNLOCK\***

Déverrouiller un message précédemment verrouillé.

**MQC.MQGMO\_WAIT**

Attendez qu'un message arrive.

**public string ResolvedQueueName {get;}**

Le gestionnaire de files d'attente définit ResolvedQueueNom sur le nom local de la file d'attente à partir de laquelle le message a été extrait. ResolvedQueueNom est différent du nom utilisé pour ouvrir la file d'attente si une file d'attente alias ou modèle a été ouverte.

**public char Segmentation {get;}\***

Segmentation indique si vous pouvez autoriser la segmentation pour le message extrait. Les valeurs possibles sont les suivantes :

**MQC.MQSEG\_INHIBITED**

Ne pas autoriser la segmentation.

**MQC.MQSEG\_ALLOWED**

Autoriser la segmentation

**public byte SegmentStatus {get;}\***

SegmentStatus est une zone de sortie qui indique si le message extrait est un segment d'un message logique. Si le message est un segment, l'indicateur indique s'il s'agit du dernier segment. Les valeurs possibles sont les suivantes :

**MQC.MQSS\_LAST\_SEGMENT**

Le message est le dernier ou le seul segment du message logique.

**MQC.MQSS\_NOT\_A\_SEGMENT**

Le message n'est pas un segment.

**MQC.MQSS\_SEGMENT**

Le message est un segment, mais n'est pas le dernier segment du message logique.

**public int WaitInterval {get; set;}**

WaitInterval est la durée maximale en millisecondes pendant laquelle un appel MQQueue.get attend l'arrivée d'un message approprié. Utilisez WaitInterval avec MQC.MQGMO\_WAIT. Définissez la valeur MQC.MQWI\_UNLIMITED pour attendre un temps illimité pour un message.

## Constructeurs

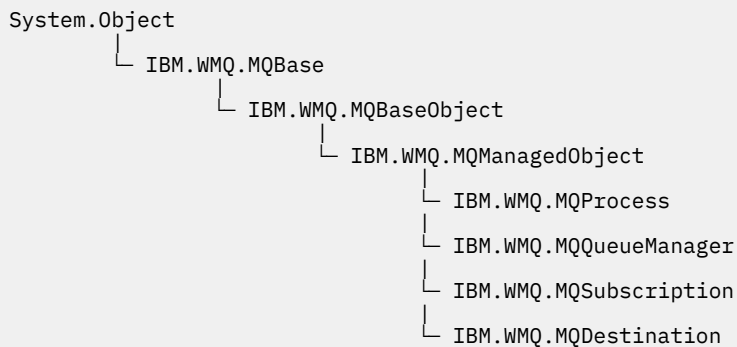
**public MQGetMessageOptions()**

Construisez un nouvel objet MQGetMessageOptions avec Options défini sur MQC.MQGMO\_NO\_WAIT, WaitInterval défini sur zéro et ResolvedQueueName défini sur vide.

## Classe MQManagedObject.NET

Utilisez MQManagedObject pour interroger et définir les attributs de MQDestination, MQProcess, MQQueueManageret MQSubscription. MQManagedObject est une superclasse de ces classes.

## Classes



```
public class IBM.WMQ.MQManagedObject extends IBM.WMQ.MQBaseObject;
```

- «Propriétés», à la page [1816](#)
- «Méthodes», à la page [1817](#)
- «Constructeurs», à la page [1818](#)

## Propriétés

Test de l'émission de MQException lors de l'obtention des propriétés.

```
public string AlternateUserId {get; set;}
```

ID utilisateur de remplacement, le cas échéant, défini lors de l'ouverture de la ressource.

AlternateUserID.set est ignoré lorsqu'il est émis pour un objet ouvert. L'IDAlternateUser n'est pas valide pour les abonnements.

```
public int CloseOptions {get; set;}
```

Définissez cet attribut pour contrôler la manière dont la ressource est fermée. La valeur par défaut est MQC.MQCO\_NONE. MQC.MQCO\_NONE est la seule valeur autorisée pour toutes les ressources autres que les files d'attente dynamiques permanentes, les files d'attente dynamiques temporaires, les abonnements et les rubriques accessibles par les objets qui les ont créées.

Pour les files d'attente et les rubriques, les valeurs supplémentaires suivantes sont autorisées:

### **MQC.MQCO\_DELETE**

Supprimez la file d'attente s'il n'y a pas de message.

### **MQC.MQCO\_DELETE\_PURGE**

Supprimez la file d'attente, en purgeant les messages qu'elle contient.

### **MQC.MQCO\_QUIESCE**

Demandez la fermeture de la file d'attente, en recevant un avertissement si des messages subsistent (ce qui permet de les extraire avant la fermeture finale).

Pour les abonnements, les valeurs supplémentaires suivantes sont autorisées:

### **MQC.MQCO\_KEEP\_SUB**

L'abonnement n'est pas supprimé. Cette option est valide uniquement si l'abonnement d'origine est durable. MQC.MQCO\_KEEP\_SUB est la valeur par défaut d'une rubrique durable.

### **MQC.MQCO\_REMOVE\_SUB**

L'abonnement est supprimé. MQC.MQCO\_REMOVE\_SUB est la valeur par défaut pour une rubrique non durable et non gérée.

### **MQC.MQCO\_PURGE\_SUB**

L'abonnement est supprimé. MQC.MQCO\_PURGE\_SUB est la valeur par défaut pour une rubrique gérée non durable.

```
public MQQueueManager ConnectionReference {get;}
```

Gestionnaire de files d'attente auquel appartient cette ressource.



**public string MQDescription {get;}**

Description de la ressource telle qu'elle est conservée par le gestionnaire de files d'attente. MQDescription renvoie une chaîne vide pour les abonnements et les rubriques.

**public boolean IsOpen {get;}**

Indique si la ressource est actuellement ouverte.

**public string Name {get;}**

Nom de la ressource. Le nom est soit celui indiqué dans la méthode d'accès, soit celui attribué par le gestionnaire de files d'attente à une file d'attente dynamique.

**public int OpenOptions {get; set;}**

Les OpenOptions sont définies lorsqu'un objet IBM MQ est ouvert. La méthode OpenOptions.set est ignorée et ne génère pas d'erreur. Les abonnements n'ont pas d'OpenOptions.

## Méthodes

**public virtual void Close();**

Emet MQException.

Ferme l'objet. Aucune autre opération sur cette ressource n'est autorisée après l'appel de Close. Pour modifier le comportement de la méthode Close, définissez l'attribut closeOptions.

**public string GetAttributeString(int selector, int length);**

Emet MQException.

Obtient une chaîne d'attribut.

**selector**

Entier indiquant l'attribut demandé.

**length**

Entier indiquant la longueur de la chaîne requise.

**public void Inquire(int[] selectors, int[] intAttrs, byte[] charAttrs);**

Emet MQException.

Renvoie un tableau d'entiers et un ensemble de chaînes de caractères contenant les attributs d'une file d'attente, d'un processus ou d'un gestionnaire de files d'attente. Les attributs à interroger sont spécifiés dans le tableau des sélecteurs.

**Remarque :** La plupart des attributs les plus courants peuvent être interrogés à l'aide des méthodes Get définies dans MQManagedObject, MQQueue et MQQueueManager.

**sélecteurs**

Tableau d'entiers identifiant les attributs avec les valeurs à identifier.

**intAttrs**

Tableau dans lequel les valeurs d'attribut entières sont renvoyées. Les valeurs d'attribut de type entier sont renvoyées dans le même ordre que les sélecteurs d'attribut de type entier dans le tableau des sélecteurs.

**charAttrs**

Mémoire tampon dans laquelle les attributs de caractères sont renvoyés, concaténée. Les attributs de caractère sont renvoyés dans le même ordre que les sélecteurs d'attribut de caractère dans le tableau des sélecteurs. La longueur de chaque chaîne d'attribut est fixe pour chaque attribut.

**public void Set(int[] selectors, int[] intAttrs, byte[] charAttrs);**

Emet MQException.

Définit les attributs définis dans le vecteur des sélecteurs. Les attributs à définir sont spécifiés dans le tableau des sélecteurs.

**sélecteurs**

Tableau d'entiers identifiant les attributs avec les valeurs à définir.

### **intAttrs**

Tableau de valeurs d'attribut entières à définir. Ces valeurs doivent être dans le même ordre que les sélecteurs d'attribut d'entier dans le tableau des sélecteurs.

### **charAttrs**

Mémoire tampon dans laquelle les attributs de caractères à définir sont concaténés. Ces valeurs doivent être dans le même ordre que les sélecteurs d'attribut de caractères dans le tableau des sélecteurs. La longueur de chaque attribut de caractère est fixe.

**public void SetAttributeString(int selector, string value, int length);**

Emet MQException.

Définit une chaîne d'attribut.

### **selector**

Entier indiquant l'attribut en cours de définition.

### **valeur**

Chaîne à définir comme valeur d'attribut.

### **length**

Entier indiquant la longueur de la chaîne requise.

## **Constructeurs**

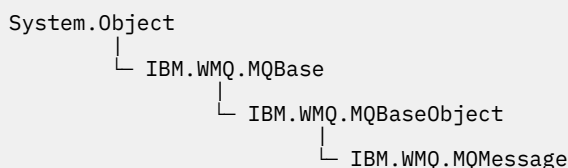
**protected MQManagedObject()**

Méthode du constructeur. Cet objet est une classe de base abstraite qui ne peut pas être instanciée par elle-même.

## **Classe MQMessage.NET**

Utilisez MQMessage pour accéder au descripteur de message et aux données d'un message IBM MQ . MQMessage encapsule un message IBM MQ .

### **Classe**



```
public class IBM.WMQ.MQMessage extends IBM.WMQ.MQBaseObject;
```

Créez un objet MQMessage , puis utilisez les méthodes Read et Write pour transférer des données entre le message et d'autres objets de votre application. Envoyez et recevez des objets MQMessage à l'aide des méthodes Put et Get des classes MQDestination, MQQueue et MQTopic .

Obtenez et définissez les propriétés du descripteur de message à l'aide des propriétés de MQMessage. Définissez et obtenez les propriétés de message étendues à l'aide des méthodes SetProperty et GetProperty .

- [«Propriétés», à la page 1819](#)
- [«Méthodes de message Read et Write», à la page 1824](#)
- [«Méthodes de mémoire tampon», à la page 1827](#)
- [«Méthodes de propriété», à la page 1828](#)
- [«Constructeurs», à la page 1830](#)

## Propriétés

Test de l'émission de `MQException` lors de l'obtention des propriétés.

### **public string AccountingToken {get; set;}**

Partie du contexte d'identité du message ; elle aide une application à facturer le travail effectué suite au message. La valeur par défaut est `MQC.MQACT_NONE`.

### **public string ApplicationIdData {get; set;}**

Partie du contexte d'identité du message. Les données `ApplicationId` sont des informations définies par la suite d'applications et peuvent être utilisées pour fournir des informations supplémentaires sur le message ou son émetteur. La valeur par défaut est "".

### **public string ApplicationOriginData {get; set;}**

Informations définies par l'application qui peuvent être utilisées pour fournir des informations supplémentaires sur l'origine du message. La valeur par défaut est "".

### **public int BackoutCount {get;}**

Nombre de fois où le message a été précédemment renvoyé et annulé par un appel `MQQueue.Get` dans le cadre d'une unité de travail. La valeur par défaut est zéro.

### **public int CharacterSet {get; set;}**

Identificateur de jeu de caractères codés des données de type caractères dans le message.

Définissez `CharacterSet` pour identifier le jeu de caractères des données de type caractère dans le message. Obtenez `CharacterSet` pour savoir dans quel jeu de caractères a été utilisé pour coder les données de type caractère dans le message.

Les applications .NET s'exécutent toujours en Unicode, alors que dans d'autres environnements, les applications s'exécutent dans le même jeu de caractères que le gestionnaire de files d'attente.

Les méthodes `ReadString` et `ReadLine` convertissent les données de type caractères du message en Unicode pour vous.

La méthode `WriteString` convertit le format Unicode au jeu de caractères codé dans `CharacterSet`. Si `CharacterSet` est défini sur sa valeur par défaut, `MQC.MQCCSI_Q_MGR`, qui est 0, aucune conversion n'a lieu et `CharacterSet` est défini sur 1200. Si vous affectez à `CharacterSet` une autre valeur, `WriteString` convertit Unicode en valeur de remplacement.

**Remarque :** Les autres méthodes de lecture et d'écriture n'utilisent pas `CharacterSet`.

- `ReadChar` et `WriteChar` lisent et écrivent un caractère Unicode dans et depuis la mémoire tampon de messages sans conversion.
- `ReadUTF` et `WriteUTF` convertissent entre une chaîne Unicode dans l'application et une chaîne UTF-8, préfixée par une zone de longueur de 2 octets, dans la mémoire tampon de messages.
- Les méthodes `Byte` transfèrent des octets entre l'application et la mémoire tampon de message sans modification.

### **public byte[] CorrelationId {get; set;}**

- Pour un appel `MQQueue.Get`, identificateur de corrélation du message à extraire. Le gestionnaire de files d'attente renvoie le premier message avec un identificateur de message et un identificateur de corrélation qui correspondent aux zones de descripteur de message. La valeur par défaut, `MQC.MQCI_NONE`, permet à tout identificateur de corrélation de correspondre.
- Pour un appel `MQQueue.Put`, identificateur de corrélation à définir.

### **public int DataLength {get;}**

Nombre d'octets de données de message restant à lire.

### **public int DataOffset {get; set;}**

Position actuelle du curseur dans les données du message. Les lectures et les écritures prennent effet à la position en cours.

## **public int Encoding {get; set;}**

Représentation utilisée pour les valeurs numériques dans les données de message d'application. Codage s'applique aux données binaires, décimales condensées et à virgule flottante. Le comportement des méthodes de lecture et d'écriture pour ces formats numériques est modifié en conséquence. Construisez une valeur pour la zone de codage en ajoutant une valeur à partir de chacune de ces trois sections. Vous pouvez également construire la valeur en combinant les valeurs de chacune des trois sections à l'aide de l'opérateur OR bit à bit.

### 1. Binary Integer

#### **MQC.MQENC\_INTEGER\_NORMAL**

Entiers big endian.

#### **MQC.MQENC\_INTEGER\_REVERSED**

Entiers little endian, tels qu'ils sont utilisés dans l'architecture Intel .

### 2. Décimal condensé

#### **MQC.MQENC\_DECIMAL\_NORMAL**

Format décimal condensé Big-endian, tel qu'il est utilisé par z/OS.

#### **MQC.MQENC\_DECIMAL\_REVERSED**

Décimale condensée little endian.

### 3. virgule flottante

#### **MQC.MQENC\_FLOAT\_IEEE\_NORMAL**

L'IEEE Big-endian flotte.

#### **MQC.MQENC\_FLOAT\_IEEE\_REVERSED**

L'IEEE little endian flotte, comme l'architecture Intel utilisée.

#### **MQC.MQENC\_FLOAT\_S390**

z/OS formate les nombres à virgule flottante.

La valeur par défaut est :

```
MQC.MQENC_INTEGER_REVERSED |
MQC.MQENC_DECIMAL_REVERSED |
MQC.MQENC_FLOAT_IEEE_REVERSED
```

Le paramètre par défaut permet à `WriteInt` d'écrire un entier little-endian et à `ReadInt` de lire un entier little-endian. Si vous définissez l'indicateur `MQC.MQENC_INTEGER_NORMAL` à la place, `WriteInt` écrit un entier big-endian et `ReadInt` lit un entier big-endian.

**Remarque :** Une perte de précision peut se produire lors de la conversion de points flottants au format IEEE en points flottants au format `zSeries` .

## **public int Expiry {get; set;}**

Heure d'expiration exprimée en dixièmes de seconde, définie par l'application qui insère le message. Une fois que le délai d'expiration d'un message est écoulé, il peut être supprimé par le gestionnaire de files d'attente. Si le message a spécifié l'un des indicateurs `MQC.MQRO_EXPIRATION` , un rapport est généré lorsque le message est supprimé. La valeur par défaut est `MQC.MQEI_UNLIMITED`, ce qui signifie que le message n'expire jamais.

## **public int Feedback {get; set;}**

Utilisez `Commentaires` avec un message de type `MQC.MQMT_REPORT` pour indiquer la nature du rapport. Les codes retour suivants sont définis par le système:

- `MQC.MQFB_EXPIRATION`
- `MQC.MQFB_COA`
- `MQC.MQFB_COD`
- `MQC.MQFB_QUIT`
- `MQC.MQFB_PAN`
- `MQC.MQFB_NAN`

- MQC.MQFB\_DATA\_LENGTH\_ZERO
- MQC.MQFB\_DATA\_LENGTH\_NEGATIVE
- MQC.MQFB\_DATA\_LENGTH\_TOO\_BIG
- MQC.MQFB\_BUFFER\_OVERFLOW
- MQC.MQFB\_LENGTH\_OFF\_BY\_ONE
- MQC.MQFB\_IIH\_ERROR

Les valeurs de retour d'informations définies par l'application dans la plage MQC.MQFB\_APPL\_FIRST à MQC.MQFB\_APPL\_LAST peuvent également être utilisées. La valeur par défaut de cette zone est MQC.MQFB\_NONE, ce qui indique qu'aucun commentaire n'est fourni.

**public string Format {get; set;}**

Nom de format utilisé par l'expéditeur du message pour indiquer la nature des données du message au destinataire. Vous pouvez utiliser vos propres noms de format, mais les noms commençant par les lettres MQ ont une signification définie par le gestionnaire de files d'attente. Les formats intégrés du gestionnaire de files d'attente sont les suivants:

**MQC.MQFMT\_ADMIN**

Message de demande / réponse du serveur de commandes.

**MQC.MQFMT\_COMMAND\_1**

Message de réponse à la commande de type 1.

**MQC.MQFMT\_COMMAND\_2**

Message de réponse à la commande de type 2.

**MQC.MQFMT\_DEAD\_LETTER\_HEADER**

En-tête de rebut.

**MQC.MQFMT\_EVENT**

Message d'événement.

**MQC.MQFMT\_NONE**

Aucun nom de format.

**MQC.MQFMT\_PCF**

Message défini par l'utilisateur au format de commande programmable.

**MQC.MQFMT\_STRING**

Message composé entièrement de caractères.

**MQC.MQFMT\_TRIGGER**

Message de déclenchement

**MQC.MQFMT\_XMIT\_Q\_HEADER**

En-tête de file d'attente de transmission.

La valeur par défaut est MQC.MQFMT\_NONE.

**public byte[] GroupId {get; set;}**

Chaîne d'octets qui identifie le groupe de messages auquel appartient le message physique. La valeur par défaut est MQC.MQGI\_NONE.

**public int MessageFlags {get; set;}**

Indicateurs contrôlant la segmentation et l'état d'un message.

**public byte[] MessageId {get; set;}**

Pour un appel MQQueue.Get, cette zone indique l'identificateur du message à extraire. Normalement, le gestionnaire de files d'attente renvoie le premier message avec un identificateur de message et un identificateur de corrélation qui correspondent aux zones de descripteur de message. Autorisez la mise en correspondance de tout identificateur de message à l'aide de la valeur spéciale MQC.MQMI\_NONE.

Pour un appel MQQueue.Put, cette zone indique l'identificateur de message à utiliser. Si MQC.MQMI\_NONE est spécifié, le gestionnaire de files d'attente génère un identificateur de message

unique lors de l'insertion du message. La valeur de cette variable de membre est mise à jour après l'insertion, pour indiquer l'identificateur de message utilisé. La valeur par défaut est MQC.MQMI\_NONE.

**public int MessageLength {get;}**

Nombre d'octets de données de message dans l'objet MQMessage .

**public int MessageSequenceNumber {get; set;}**

Numéro de séquence d'un message logique au sein d'un groupe.

**public int MessageType {get; set;}**

Indique le type du message. Les valeurs suivantes sont actuellement définies par le système:

- MQC.MQMT\_DATAGRAM
- MQC.MQMT\_REPLY
- MQC.MQMT\_REPORT
- MQC.MQMT\_REQUEST

Les valeurs définies par l'application peuvent également être utilisées, dans la plage MQC.MQMT\_APPL\_FIRST à MQC.MQMT\_APPL\_LAST. La valeur par défaut de cette zone est MQC.MQMT\_DATAGRAM.

**public int Offset {get; set;}**

Dans un message segmenté, décalage des données dans un message physique à partir du début d'un message logique.

**public int OriginalLength {get; set;}**

Longueur d'origine d'un message segmenté.

**public int Persistence {get; set;}**

Persistence des messages. Les valeurs suivantes sont définies :

- MQC.MQPER\_NOT\_PERSISTENT

Si vous définissez cette option dans un client reconnectable, le code anomalie MQRC\_NONE est renvoyé à l'application lorsque la connexion aboutit.

- MQC.MQPER\_PERSISTENT

Si vous définissez cette option dans un client reconnectable, le code anomalie MQRC\_CALL\_INTERRUPTED est renvoyé à l'application une fois la connexion établie.

- MQC.MQPER\_PERSISTENCE\_AS\_Q\_DEF

La valeur par défaut est MQC.MQPER\_PERSISTENCE\_AS\_Q\_DEF, qui extrait la persistance du message de l'attribut de persistance par défaut de la file d'attente de destination.

**public int Priority {get; set;}**

Priorité du message. La valeur spéciale MQC.MQPRI\_PRIORITY\_AS\_Q\_DEF peut également être définie dans le message sortant. La priorité du message est ensuite extraite de l'attribut de priorité par défaut de la file d'attente de destination. La valeur par défaut est MQC.MQPRI\_PRIORITY\_AS\_Q\_DEF.

**public int PropertyValidation {get; set;}**

Indique si la validation des propriétés a lieu lorsqu'une propriété du message est définie. Les valeurs possibles sont les suivantes :

- MQCMHO\_DEFAULT\_VALIDATION
- MQCMHO\_VALIDATE
- MQCMHO\_NO\_VALIDATION

La valeur par défaut est MQCMHO\_DEFAULT\_VALIDATION.

**public string PutApplicationName {get; set;}**

Nom de l'application qui a inséré le message. La valeur par défaut est " " .

**public int PutApplicationType {get; set;}**

Type de l'application ayant placé le message en file d'attente. PutApplicationType peut être une valeur définie par le système ou par l'utilisateur. Les valeurs suivantes sont définies par le système:

- MQC.MQAT\_AIX
- MQC.MQAT\_CICS
- MQC.MQAT\_DOS
- MQC.MQAT\_IMS
- MQC.MQAT\_MVS
- MQC.MQAT\_OS2
- MQC.MQAT\_OS400
- MQC.MQAT\_QMGR
- MQC.MQAT\_UNIX
- MQC.MQAT\_WINDOWS
- MQC.MQAT\_JAVA

La valeur par défaut est MQC.MQAT\_NO\_CONTEXT, ce qui indique qu'aucune information contextuelle n'est présente dans le message.

**public DateTime PutDateTime {get; set;}**

Date et heure à laquelle le message a été inséré.

**public string ReplyToQueueManagerName {get; set;}**

Nom du gestionnaire de files d'attente pour l'envoi des messages de réponse ou de rapport. La valeur par défaut est "" et le gestionnaire de files d'attente fournit le nom ReplyToQueueManager.

**public string ReplyToQueueName {get; set;}**

Nom de la file d'attente de messages à laquelle l'application qui a émis la demande d'obtention du message envoie des messages MQC.MQMT\_REPLY et MQC.MQMT\_REPORT. Le ReplyToQueueName par défaut est "".

**public int Report {get; set;}**

Utilisez Report pour spécifier les options relatives aux messages de rapport et de réponse:

- Indique si des rapports sont requis.
- Indique si les données de message d'application doivent être incluses dans les rapports.
- Comment définir les identificateurs de message et de corrélation dans le rapport ou la réponse.

Toute combinaison des quatre types de rapport peut être demandée:

- Indiquez une combinaison des quatre types de rapport. Sélection de l'une des trois options pour chaque type de rapport, selon que les données de message d'application doivent ou non être incluses dans le message de rapport.

1. Confirmer à réception

- MQC.MQRO\_COA
- MQC.MQRO\_COA\_WITH\_DATA
- MQC.MQRO\_COA\_WITH\_FULL\_DATA \*\*

2. Confirmer à expédition

- MQC.MQRO\_COD
- MQC.MQRO\_COD\_WITH\_DATA
- MQC.MQRO\_COD\_WITH\_FULL\_DATA \*\*

3. Exception

- MQC.MQRO\_EXCEPTION
- MQC.MQRO\_EXCEPTION\_WITH\_DATA

– MQC.MQRO\_EXCEPTION\_WITH\_FULL\_DATA \*\*

#### 4. Expiration

– MQC.MQRO\_EXPIRATION

– MQC.MQRO\_EXPIRATION\_WITH\_DATA

– MQC.MQRO\_EXPIRATION\_WITH\_FULL\_DATA \*\*

**Remarque :** Les valeurs signalées par \*\* dans la liste ne sont pas prises en charge par les gestionnaires de files d'attente z/OS . Ne les utilisez pas si votre application est susceptible d'accéder à un gestionnaire de files d'attente z/OS , quelle que soit la plateforme sur laquelle l'application s'exécute.

- Indiquez l'une des options suivantes pour contrôler la façon dont l'ID message est généré pour le message de rapport ou de réponse:
  - MQC.MQRO\_NEW\_MSG\_ID
  - MQC.MQRO\_PASS\_MSG\_ID
- Indiquez l'une des options suivantes pour contrôler la façon dont l'ID de corrélation du message de rapport ou de réponse doit être défini:
  - MQC.MQRO\_COPY\_MSG\_ID\_TO\_CORREL\_ID
  - MQC.MQRO\_PASS\_CORREL\_ID
- Indiquez l'une des options suivantes pour contrôler la disposition du message d'origine lorsqu'il ne peut pas être distribué à la file d'attente de destination:
  - MQC.MQRO\_DEAD\_LETTER\_Q
  - MQC.MQRO\_DISCARD\_MSG \*\*
- Si aucune option de rapport n'est spécifiée, la valeur par défaut est:

```
MQC.MQRO_NEW_MSG_ID |
MQC.MQRO_COPY_MSG_ID_TO_CORREL_ID |
MQC.MQRO_DEAD_LETTER_Q
```

- Vous pouvez spécifier l'un des éléments suivants ou les deux pour demander à l'application de réception d'envoyer un message de rapport d'action positive ou négative.
  - MQC.MQRO\_PAN
  - MQC.MQRO\_NAN

#### **public int TotalMessageLength {get;}**

Nombre total d'octets du message tels qu'ils sont stockés dans la file d'attente de messages à partir de laquelle ce message a été reçu.

#### **public string UserId {get; set;}**

UserId fait partie du contexte d'identité du message. Le gestionnaire de files d'attente fournit généralement la valeur. Vous pouvez remplacer la valeur si vous avez le droit de définir le contexte d'identité.

#### **public int Version {get; set;}**

Version de la structure MQMD utilisée.

## **Méthodes de message Read et Write**

Les méthodes Read et Write exécutent les mêmes fonctions que les membres des classes BinaryReader et BinaryWriter dans l'espace de nom .NET System.IO . Voir MSDN pour obtenir des exemples complets de syntaxe et d'utilisation du langage. Les méthodes lisent ou écrivent à partir de la position en cours dans la mémoire tampon du message. Ils déplacent la position actuelle vers l'avant en fonction du nombre d'octets lus ou écrits.

**Remarque :** Si les données de message contiennent un en-tête MQRFH ou MQRFH2 , vous devez utiliser la méthode ReadBytes pour lire les données.



- Toutes les méthodes émettent `IOException`.
- Les méthodes `ReadFully` redimensionnent automatiquement le tableau `byte` ou `sbyte` cible pour qu'il corresponde exactement au message. Un tableau `null` est également redimensionné.
- Les méthodes `Read` émettent `EndOfStreamException`.
- Les méthodes `WriteDecimal` émettent `MQException`.
- Les méthodes `ReadString`, `ReadLine` et `WriteString` sont converties entre `Unicode` et le jeu de caractères du message ; voir `CharacterSet`.
- Les méthodes `Decimal` lisent et écrivent des nombres décimaux condensés codés au format big-endian, `MQC.MQENC_DECIMAL_NORMAL` ou little-endian `MQC.MQENC_DECIMAL_REVERSE`, en fonction de la valeur de `Encoding`. Les plages décimales et les types `.NET` correspondants sont les suivants:

**Decimal2/short**

-999 à 999

**Decimal4/int**

-9999999 à 9999999

**Decimal8/long**

-9999999999999999 à 9999999999999999

- Les méthodes `Double` et `Float` lisent et écrivent des valeurs flottantes codées aux formats IEE big-endian et little-endian, `MQC.MQENC_FLOAT_IEEE_NORMAL` et `MQC.MQENC_FLOAT_IEEE_REVERSED`, ou au format `S/390`, `MQC.MQENC_FLOAT_S390`, en fonction de la valeur de `Encoding`.
- Les méthodes `Int` lisent et écrivent des valeurs entières codées au format big-endian, `MQC.MQENC_INTEGER_NORMAL` ou little-endian, `MQC.MQENC_INTEGER_REVERSED`, en fonction de la valeur de `Encoding`. Les entiers sont tous signés, à l'exception de l'ajout d'un type entier de 2 octets non signé. Les tailles d'entier et les types `.NET` et `IBM MQ` sont les suivants:

**2 octets**

`short`, `Int2`, `ushort`, `UInt2`

**4 octets**

`int`, `Int4`

**8 octets**

`long`, `Int8`

- `WriteObject` transfère la classe d'un objet, les valeurs de ses zones non transitoires et non statiques et les zones de ses supertypes, dans la mémoire tampon de messages.
- `ReadObject` crée un objet à partir de la classe de l'objet, de la signature de la classe et des valeurs de ses zones non transitoires et non statiques, ainsi que des zones de ses supertypes.

Tableau 843. Méthodes de lecture et d'écriture de messages	
Type de cible	Signatures de méthode
<b>Boolean</b>	<pre>public bool ReadBoolean();  public void WriteBoolean(bool value);</pre>
<b>Byte</b>	<pre>public byte ReadByte() public byte ReadUnsignedByte()  public void Write(int value) public void WriteByte(int value) public void WriteByte(byte value) public void WriteByte(sbyte value)</pre>

Tableau 843. Méthodes de lecture et d'écriture de messages (suite)

Type de cible	Signatures de méthode
<b>Bytes</b>	<pre> public byte[] ReadBytes(int count) public void ReadFully(ref byte[] value) public void ReadFully(ref sbyte[] value) public void ReadFully(ref byte[] value, int offset,int length) public void ReadFully(ref sbyte[] value, int offset,int length)  public void Write(byte[] value) public void Write(sbyte[] value) public void Write(byte[] value, int offset,int length) public void Write(sbyte[] value, int offset,int length) public void WriteBytes(string value)                     </pre>
<b>Decimal2</b>	<pre> public void WriteDecimal2(short value)                     </pre>
<b>Decimal4</b>	<pre> public void WriteDecimal4(short value)                     </pre>
<b>Decimal8</b>	<pre> public void WriteDecimal8(short value)                     </pre>
<b>Double</b>	<pre> public double ReadDouble()  public void WriteDouble(double value)                     </pre>
<b>Float</b>	<pre> public float ReadFloat()  public void WriteFloat(float value)                     </pre>
<b>Int2</b>	<pre> public void WriteInt2(int value)                     </pre>
<b>Int4</b>	<pre> public int readDecimal4() public int ReadInt() public int ReadInt4()  public void WriteInt(int value) public void WriteInt4(int value)                     </pre>
<b>Int8</b>	<pre> public void WriteInt8(long value)                     </pre>

Tableau 843. Méthodes de lecture et d'écriture de messages (suite)

Type de cible	Signatures de méthode
<b>Long</b>	<pre>public long ReadDecimal8() public long ReadLong() public long ReadInt8()  public void WriteLong(long value)</pre>
<b>Object</b>	<pre>public Object ReadObject()  public void WriteObject(Object object)</pre>
<b>Short</b>	<pre>public short ReadShort() public short ReadDecimal2() public short ReadInt2()  public void WriteShort(int value)</pre>
<b>string</b>	<pre>public string ReadString(int length)  public void WriteString(string string)</pre>
<b>Unsigned Short</b>	<pre>public ushort ReadUnsignedShort() public ushort ReadUInt2()</pre>
<b>Unicode</b>	<pre>public string ReadLine() public char ReadChar()  public void WriteChar(int value) public void WriteChars(string string)</pre>
<b>UTF</b>	<pre>public string ReadUTF()  public void WriteUTF(string string)</pre>

### Méthodes de mémoire tampon

**public void ClearMessage();**

Emet IOException.

Supprime toutes les données de la mémoire tampon de messages et redéfinit le décalage des données sur zéro.

### **public void ResizeBuffer(int size)**

Emet IOException.

Suggestion à l'objet MQMessage concernant la taille de la mémoire tampon pouvant être requise pour les opérations d'extraction suivantes. Si le message contient actuellement des données de message et que la nouvelle taille est inférieure à la taille en cours, les données de message sont tronquées.

### **public void Seek(int pos)**

Emet IOException, ArgumentOutOfRangeException, ArgumentException.

Déplace le curseur à la position absolue dans la mémoire tampon de messages indiquée par *pos*. Les lectures et écritures suivantes agissent à cet emplacement dans la mémoire tampon.

### **public int SkipBytes(int i)**

Emet IOException, EndOfStreamException.

Avance de *n* octets dans la mémoire tampon du message et renvoie *n*, le nombre d'octets ignorés.

La méthode SkipBytes se bloque jusqu'à ce que l'un des événements suivants se produise:

- Tous les octets sont ignorés
- La fin de la mémoire tampon de messages est détectée
- Une exception est émise

## **Méthodes de propriété**

### **public void DeleteProperty(string name);**

Emet MQException.

Supprime du message une propriété portant le nom spécifié.

#### **Nom**

Nom de la propriété à supprimer.

### **public System.Collections.IEnumerator GetPropertyNames(string name)**

Emet MQException.

Renvoie un IEnumerator de tous les noms de propriété correspondant au nom spécifié. Le signe de pourcentage '%' peut être utilisé à la fin du nom en tant que caractère générique pour filtrer les propriétés du message, correspondant à zéro ou à plusieurs caractères, y compris le point.

#### **Nom**

Nom de la propriété à mettre en correspondance.

## **Méthodes SetProperty et GetProperty**

Toutes les méthodes SetProperty et GetProperty émettent MQException.

La méthode SetProperty de la classe MQMessage .NET ajoute une nouvelle propriété si aucune propriété n'existe déjà. Toutefois, si la propriété existe déjà, la valeur de propriété fournie est ajoutée à la fin de la liste. Lorsque plusieurs valeurs sont définies sur un nom de propriété à l'aide de SetProperty, l'appel de GetProperty pour ce nom renvoie ces valeurs de manière séquentielle dans l'ordre dans lequel ces valeurs ont été définies.

Le comportement est identique pour toutes les méthodes de type Set\*Property et Get\*Property telles que GetLongProperty, SetLongProperty, GetBooleanProperty, SetBooleanProperty, GetStringProperty et SetStringProperty.

Tableau 844. Méthodes *SetProperty* et *GetProperty*

Tapez	Signatures de méthode
<b>Boolea n</b>	<pre>public boolean GetBooleanProperty(string name); public boolean GetBooleanProperty(string name, MQPropertyDescriptor pd);  public void SetBooleanProperty(string name, boolean value); public void SetBooleanProperty(string name, MQPropertyDescriptor pd, boolean value);</pre>
<b>Byte</b>	<pre>public sbyte GetByteProperty(string name); public sbyte GetByteProperty(string name, MQPropertyDescriptor pd);  public void SetByteProperty(string name, sbyte value); public void SetByteProperty(string name, MQPropertyDescriptor pd, sbyte value);</pre>
<b>Bytes</b>	<pre>public sbyte[] GetBytesProperty(string name); public sbyte[] GetBytesProperty(string name, MQPropertyDescriptor pd);  public void SetBytesProperty(string name, sbyte[] value); public void SetBytesProperty(string name, MQPropertyDescriptor pd, sbyte[] value);</pre>
<b>Double</b>	<pre>public double GetDoubleProperty(string name); public double GetDoubleProperty(string name, MQPropertyDescriptor pd);  public void SetDoubleProperty(string name, double value); public void SetDoubleProperty(string name, MQPropertyDescriptor pd, double value);</pre>
<b>Float</b>	<pre>public float GetFloatProperty(string name); public float GetFloatProperty(string name, MQPropertyDescriptor pd);  public void SetFloatProperty(string name, float value); public void SetFloatProperty(string name, MQPropertyDescriptor pd, float value);</pre>
<b>Int2</b>	<pre>public short GetInt2Property(string name); public short GetInt2Property(string name, MQPropertyDescriptor pd);  public void SetInt2Property(string name, short value); public void SetInt2Property(string name, MQPropertyDescriptor pd, short value);</pre>
<b>Int4</b>	<pre>public int GetInt4Property(string name); public int GetInt4Property(string name, MQPropertyDescriptor pd);  public void SetInt4Property(string name, int value); public void SetInt4Property(string name, MQPropertyDescriptor pd, int value);</pre>

Tableau 844. Méthodes SetProperty et GetProperty (suite)

Tapez	Signatures de méthode
<b>Int8</b>	<pre>public long GetInt8Property(string name); public long GetInt8Property(string name, MQPropertyDescriptor pd);  public void SetInt8Property(string name, long value); public void SetInt8Property(string name, MQPropertyDescriptor pd, long value);</pre>
<b>Long</b>	<pre>public long GetLongProperty(string name); public long GetLongProperty(string name, MQPropertyDescriptor pd);  public void SetLongProperty(string name, long value); public void SetLongProperty(string name, MQPropertyDescriptor pd, long value);</pre>
<b>Object</b>	<pre>public Object GetObjectProperty(string name); public Object GetObjectProperty(string name, MQPropertyDescriptor pd);  public void SetObjectProperty(string name, Object value); public void SetObjectProperty(string name, MQPropertyDescriptor pd, Object value);</pre>
<b>Short</b>	<pre>public short GetShortProperty(string name); public short GetShortProperty(string name, MQPropertyDescriptor pd);  public void SetShortProperty(string name, short value); public void SetShortProperty(string name, MQPropertyDescriptor pd, short value);</pre>
<b>string</b>	<pre>public string GetStringProperty(string name); public string GetStringProperty(string name, MQPropertyDescriptor pd);  public void SetStringProperty(string name, string value); public void SetStringProperty(string name, MQPropertyDescriptor pd, string value);</pre>

## Constructeurs

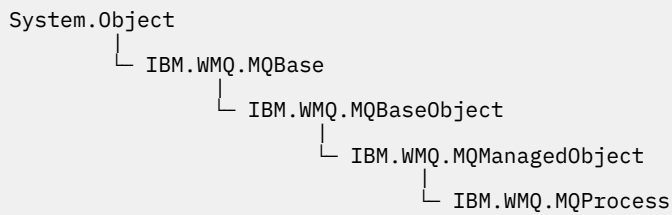
**public MQMessage();**

Crée un objet MQMessage avec des informations de descripteur de message par défaut et une mémoire tampon de message vide.

## Classe MQProcess.NET

Utilisez MQProcess pour interroger les attributs d'un processus IBM MQ . Créez un objet MQProcess à l'aide d'un constructeur ou d'une méthode MQQueueManager AccessProcess .

## Classe



```
public class IBM.WMQ.MQProcess extends IBM.WMQ.MQManagedObject;
```

- «Propriétés», à la page 1831
- «Constructeurs», à la page 1832

## Propriétés

Test de l'émission de MQException lors de l'obtention des propriétés.

```
public string ApplicationId {get;}
```

Extrait la chaîne de caractères qui identifie l'application à démarrer. ApplicationId est utilisé par une application de moniteur de déclenchement. ApplicationId est envoyé à la file d'attente d'initialisation dans le cadre du message de déclenchement.

La valeur par défaut est Null (indéfinie).

```
public int ApplicationType {get;}
```

Identifie le type de processus à démarrer par une application de moniteur de déclenchement. Des types standard sont définis, mais d'autres types peuvent être utilisés:

- MQAT\_AIX
- MQAT\_CICS
- MQAT\_IMS
- MQAT\_MVS
- MQAT\_NATIVE
- MQAT\_OS400
- MQAT\_UNIX
- MQAT\_WINDOWS
- MQAT\_JAVA
- MQAT\_USER\_FIRST
- MQAT\_USER\_LAST

La valeur par défaut est MQAT\_NATIVE.

```
public string EnvironmentData {get;}
```

Permet d'obtenir des informations sur l'environnement de l'application à démarrer.

La valeur par défaut est Null (indéfinie).

```
public string UserData {get;}
```

Obtient les informations fournies par l'utilisateur sur l'application à démarrer.

La valeur par défaut est Null (indéfinie).

## Constructeurs

```
public MQProcess(MQQueueManager queueManager, string processName, int openOptions);  
public MQProcess(MQQueueManager qMgr, string processName, int openOptions,  
string queueManagerName, string alternateUserId);
```

Emet MQException.

Accédez à un processus IBM MQ sur le gestionnaire de files d'attente *qMgr* pour consulter les attributs de processus.

### **qMgr**

Gestionnaire de files d'attente auquel accéder.

### **processName**

Nom du processus à ouvrir.

### **openOptions**

Options qui contrôlent l'ouverture du processus. Les options valides qui peuvent être ajoutées ou combinées à l'aide d'un OR bit à bit sont les suivantes:

- MQC.MQ00\_FAIL\_IF QUIESCING
- MQC.MQ00\_INQUIRE
- MQC.MQ00\_SET
- MQC.MQ00\_ALTERNATE\_USER\_AUTHORITY

### **QueueManagerName**

Nom du gestionnaire de files d'attente sur lequel le processus est défini. Vous pouvez laisser un nom de gestionnaire de files d'attente vide ou null si le gestionnaire de files d'attente est identique à celui auquel le processus accède.

### **AlternateUserid**

Si MQC.MQ00\_ALTERNATE\_USER\_AUTHORITY est spécifié dans le paramètre **openOptions**, *alternateUserId* indique l'autre ID utilisateur utilisé pour vérifier l'autorisation de l'action. Si MQ00\_ALTERNATE\_USER\_AUTHORITY n'est pas spécifié, *alternateUserId* peut être vide ou null.

Les droits utilisateur par défaut sont utilisés pour la connexion au gestionnaire de files d'attente si MQC.MQ00\_ALTERNATE\_USER\_AUTHORITY n'est pas spécifié.

```
public MQProcess MQQueueManager.AccessProcess(string processName, int openOptions);  
public MQProcess MQQueueManager.AccessProcess(string processName, int openOptions,  
string queueManagerName, string alternateUserId);
```

Emet MQException.

Accédez à un processus IBM MQ sur ce gestionnaire de files d'attente pour consulter les attributs de processus.

### **processName**

Nom du processus à ouvrir.

### **openOptions**

Options qui contrôlent l'ouverture du processus. Les options valides qui peuvent être ajoutées ou combinées à l'aide d'un OR bit à bit sont les suivantes:

- MQC.MQ00\_FAIL\_IF QUIESCING
- MQC.MQ00\_INQUIRE
- MQC.MQ00\_SET
- MQC.MQ00\_ALTERNATE\_USER\_AUTHORITY



### QueueManagerName

Nom du gestionnaire de files d'attente sur lequel le processus est défini. Vous pouvez laisser un nom de gestionnaire de files d'attente vide ou null si le gestionnaire de files d'attente est identique à celui auquel le processus accède.

### AlternateUserid

Si MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY est spécifié dans le paramètre **openOptions**, *alternateUserId* indique l'autre ID utilisateur utilisé pour vérifier l'autorisation de l'action. Si MQOO\_ALTERNATE\_USER\_AUTHORITY n'est pas spécifié, *alternateUserId* peut être vide ou null.

Les droits utilisateur par défaut sont utilisés pour la connexion au gestionnaire de files d'attente si MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY n'est pas spécifié.

## Classe MQPropertyDescriptor.NET

Utilisez MQPropertyDescriptor comme paramètre pour les méthodes MQMessage GetProperty et SetProperty. MQPropertyDescriptor décrit une propriété MQMessage.

### Classe

```
System.Object
└─ IBM.WMQ.MQPropertyDescriptor
```

```
public class IBM.WMQ.MQPropertyDescriptor extends System.Object;
```

- «Propriétés», à la page 1833
- «Constructeurs», à la page 1834

### Propriétés

Test de l'émission de MQException lors de l'obtention des propriétés.

```
public int Context {get; set;}
```

Contexte de message auquel appartient la propriété. Les valeurs possibles sont les suivantes :

#### MQC.MQPD\_NO\_CONTEXT

La propriété n'est pas associée à un contexte de message.

#### MQC.MQPD\_USER\_CONTEXT

La propriété est associée au contexte utilisateur.

Si l'utilisateur est autorisé, une propriété associée au contexte utilisateur est sauvegardée lorsqu'un message est extrait. Une méthode Put ultérieure référençant le contexte sauvegardé peut transmettre la propriété dans le nouveau message.

```
public int CopyOptions {get; set;}
```

CopyOptions décrit le type de message dans lequel la propriété peut être copiée.

Lorsqu'un gestionnaire de files d'attente reçoit un message contenant une propriété définie par IBM MQ qu'il reconnaît comme incorrecte, il corrige la valeur de la zone CopyOptions.

Vous pouvez indiquer n'importe quelle combinaison des options suivantes. Combinez les options en ajoutant les valeurs ou en utilisant le ORbit à bit.

#### MQC.MQCOPY\_ALL

La propriété est copiée dans tous les types de messages suivants.

#### MQC.MQCOPY\_FORWARD

La propriété est copiée dans un message en cours de transfert.

**MQC.MQCOPY\_PUBLISH**

La propriété est copiée dans le message reçu par un abonné lorsqu'un message est publié.

**MQC.MQCOPY\_REPLY**

La propriété est copiée dans un message de réponse.

**MQC.MQCOPY\_REPORT**

La propriété est copiée dans un message de rapport.

**MQC.MQCOPY\_DEFAULT**

La valeur indiquée indique qu'aucune autre option de copie n'a été spécifiée. Il n'existe aucune relation entre la propriété et les messages suivants. MQC.MQCOPY\_DEFAULT est toujours renvoyé pour les propriétés de descripteur de message.

**MQC.MQCOPY\_NONE**

Identique à MQC.MQCOPY\_DEFAULT

```
public int Options { set; }
```

Options prend par défaut la valeur CMQC.MQPD\_NONE. Vous ne pouvez pas définir d'autre valeur.

```
public int Support { get; set; }
```

Définissez Support pour spécifier le niveau de prise en charge requis pour les propriétés de message définies par IBM MQ. La prise en charge de toutes les autres propriétés est facultative. Aucune des valeurs suivantes ne peut être spécifiée

**MQC.MQPD\_SUPPORT\_OPTIONAL**

La propriété est acceptée par un gestionnaire de files d'attente même si elle n'est pas prise en charge. La propriété peut être supprimée pour que le message soit transmis à un gestionnaire de files d'attente qui ne prend pas en charge les propriétés de message. Cette valeur est également affectée aux propriétés qui ne sont pas définies par IBM MQ .

**MQC.MQPD\_SUPPORT\_REQUIRED**

La prise en charge de la propriété est requise. Si vous placez le message dans un gestionnaire de files d'attente qui ne prend pas en charge la propriété définie par IBM MQ, la méthode échoue. Elle renvoie le code achèvement MQC.MQCC\_FAILED et le code anomalie MQC.MQRC\_UNSUPPORTED\_PROPERTY.

**MQC.MQPD\_SUPPORT\_REQUIRED\_IF\_LOCAL**

La prise en charge de la propriété est requise si le message est destiné à une file d'attente locale. Si vous placez le message dans une file d'attente locale sur un gestionnaire de files d'attente qui ne prend pas en charge la propriété définie par IBM MQ, la méthode échoue. Elle renvoie le code achèvement MQC.MQCC\_FAILED et le code anomalie MQC.MQRC\_UNSUPPORTED\_PROPERTY.

Aucune vérification n'est effectuée si le message est inséré dans un gestionnaire de files d'attente éloignées.

**Constructeurs**

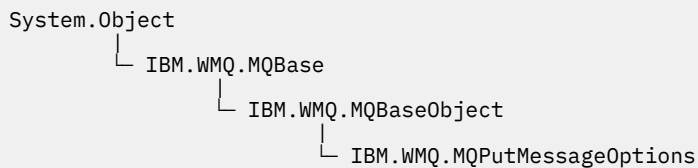
```
PropertyDescriptor();
```

Créez un descripteur de propriété.

**Classe MQPutMessageOptions.NET**

Utilisez MQPutMessageOptions pour spécifier le mode d'envoi des messages. Il modifie le comportement de MQDestination.Put.

**Classe**



```
public class IBM.WMQ.MQPutMessageOptions extends IBM.WMQ.MQBaseObject;
```

- «Propriétés», à la page 1835 «Constructeurs», à la page 1837

## Propriétés

Test de l'émission de `MQException` lors de l'obtention des propriétés.

**Remarque :** Le comportement de certaines des options disponibles dans cette classe dépend de l'environnement dans lequel elles sont utilisées. Ces éléments sont marqués d'un astérisque, \*.

**public MQQueue ContextReference {get; set;}**

Si la zone options inclut `MQC.MQPMO_PASS_IDENTITY_CONTEXT` ou `MQC.MQPMO_PASS_ALL_CONTEXT`, définissez cette zone pour qu'elle fasse référence au `MQQueue` à partir duquel les informations de contexte doivent être prises.

La valeur initiale de cette zone est null.

**public int InvalidDestCount {get;} \***

Généralement, utilisé pour les listes de distribution, `InvalidDestCount` indique le nombre de messages qui n'ont pas pu être envoyés aux files d'attente d'une liste de distribution. Le nombre inclut les files d'attente dont l'ouverture a échoué, ainsi que les files d'attente dont l'ouverture a abouti, mais pour lesquelles l'opération d'insertion a échoué.

.NET ne prend pas en charge les listes de distribution, mais `InvalidDestCount` est défini lors de l'ouverture d'une file d'attente unique.

**public int KnownDestCount {get;} \***

Généralement utilisé pour les listes de distribution, `KnownDestCount` indique le nombre de messages que l'appel en cours a envoyés avec succès à des files d'attente qui se résolvent en files d'attente locales.

.NET ne prend pas en charge les listes de distribution, mais `InvalidDestCount` est défini lors de l'ouverture d'une file d'attente unique.

**public int Options {get; set;}**

Options qui contrôlent l'action de `MQDestination.put` et de `MQQueueManager.put`. Aucune des valeurs suivantes ne peut être indiquée. Si plusieurs options sont requises, les valeurs peuvent être ajoutées ou combinées à l'aide de l'opérateur OR bit à bit.

**MQC.MQPMO\_ASYNC\_RESPONSE**

Cette option permet d'effectuer l'appel `MQDestination.put` de manière asynchrone avec des données de réponse.

**MQC.MQPMO\_DEFAULT\_CONTEXT**

Associez le contexte par défaut au message.

**MQC.MQPMO\_FAIL\_IF QUIESCING**

Echec si le gestionnaire de files d'attente est au repos.

**MQC.MQPMO\_LOGICAL\_ORDER \***

Placez les messages logiques et les segments des groupes de messages dans leur ordre logique.

Si vous utilisez l'option `MQPMO_LOGICAL_ORDER` dans un client reconnectable, le code anomalie `MQRC_RECONNECT_INCOMPATIBLE` est renvoyé à l'application.

**MQC.MQPMO\_NEW\_CORREL\_ID \***

Générez un nouvel ID de corrélation pour chaque message envoyé.

**MQC.MQPMO\_NEW\_MSG\_ID \***

Générez un nouvel ID message pour chaque message envoyé.

**MQC.MQPMO\_NONE**

Aucune option spécifiée. Ne pas utiliser avec d'autres options.

**MQC.MQPMO\_NO\_CONTEXT**

Aucun contexte ne doit être associé au message.

**MQC.MQPMO\_NO\_SYNCPOINT**

Placez un message sans contrôle de point de synchronisation. Si l'option de contrôle du point de synchronisation n'est pas spécifiée, la valeur par défaut est "aucun point de synchronisation".

**MQC.MQPMO\_PASS\_ALL\_CONTEXT**

Transmettez tout le contexte à partir d'un descripteur de file d'attente d'entrée.

**MQC.MQPMO\_PASS\_IDENTITY\_CONTEXT**

Transmettez le contexte d'identité à partir d'un descripteur de file d'attente d'entrée.

**MQC.MQPMO\_RESPONSE\_AS\_Q\_DEF**

Pour un appel `MQDestination.put`, cette option prend le type de réponse d'insertion de l'attribut `DEFPRESP` de la file d'attente.

Pour un appel `MQQueueManager.put`, cette option permet d'effectuer l'appel de manière synchrone.

**MQC.MQPMO\_RESPONSE\_AS\_TOPIC\_DEF**

`MQC.MQPMO_RESPONSE_AS_TOPIC_DEF` est un synonyme de `MQC.MQPMO_RESPONSE_AS_Q_DEF` à utiliser avec les objets de rubrique.

**MQC.MQPMO\_RETAIN**

La publication envoyée doit être conservée par le gestionnaire de files d'attente. Si cette option est utilisée et que la publication ne peut pas être conservée, le message n'est pas publié et l'appel échoue avec `MQC.MQRC_PUT_NOT_RETAINED`.

Demandez une copie de cette publication après l'heure à laquelle elle a été publiée, en appelant la méthode `MQSubscription.RequestPublicationUpdate`. La publication sauvegardée est envoyée aux applications qui créent un abonnement sans définir l'option `MQC.MQSO_NEW_PUBLICATIONS_ONLY`. Vérifiez la propriété de message `MQIsRetained` d'une publication, lorsqu'elle est reçue, pour déterminer s'il s'agit de la publication conservée.

Lorsque des publications conservées sont demandées par un abonné, l'abonnement utilisé peut contenir un caractère générique dans la chaîne de rubrique. S'il existe plusieurs publications conservées dans l'arborescence de rubriques qui correspondent à l'abonnement, elles sont toutes envoyées.

**MQC.MQPMO\_SET\_ALL\_CONTEXT**

Définissez tous les contextes à partir de l'application.

**MQC.MQPMO\_SET\_IDENTITY\_CONTEXT**

Définissez le contexte d'identité à partir de l'application.

**MQC.MQPMO\_SYNC\_RESPONSE**

Cette option permet d'effectuer l'appel `MQDestination.put` ou `MQQueueManager.put` de manière synchrone, avec des données de réponse complètes.

**MQC.MQPMO\_SUPPRESS\_REPLYTO**

Les informations renseignées dans les zones `ReplyToQueueName` et `ReplyToQueueManagerName` de la publication ne sont pas transmises aux abonnés. Si cette option est utilisée en combinaison avec une option de rapport qui requiert un `ReplyToQueueName`, l'appel échoue avec `MQC.MQRC_MISSING_REPLY_TO_Q`.

## **MQC.MQPMO\_SYNCPOINT**

Placez un message avec le contrôle de point de synchronisation. Le message n'est pas visible en dehors de l'unité de travail tant que l'unité de travail n'est pas validée. Si l'unité d'oeuvre est annulée, le message est supprimé.

### **public int RecordFields {get; set;} \***

Informations sur les listes de distribution. Les listes de distribution ne sont pas prises en charge dans .NET.

### **public string ResolvedQueueManagerName {get;}**

Zone de sortie définie par le gestionnaire de files d'attente sur le nom du gestionnaire de files d'attente propriétaire de la file d'attente spécifiée par le nom de la file d'attente éloignée. ResolvedQueueManagerName peut être différent du nom du gestionnaire de files d'attente à partir duquel la file d'attente a été consultée si la file d'attente est une file d'attente éloignée.

Une valeur non vide est renvoyée uniquement si l'objet est une file d'attente unique. Si l'objet est une liste de distribution ou une rubrique, la valeur renvoyée n'est pas définie.

### **public string ResolvedQueueName {get;}**

Zone de sortie définie par le gestionnaire de files d'attente sur le nom de la file d'attente dans laquelle le message est placé. ResolvedQueueName peut être différent du nom utilisé pour ouvrir la file d'attente si la file d'attente ouverte était un alias ou une file d'attente modèle.

Une valeur non vide est renvoyée uniquement si l'objet est une file d'attente unique. Si l'objet est une liste de distribution ou une rubrique, la valeur renvoyée n'est pas définie.

### **public int UnknownDestCount {get;} \***

Généralement utilisé pour les listes de distribution, UnknownDestCount est une zone de sortie définie par le gestionnaire de files d'attente. Il indique le nombre de messages que l'appel en cours a envoyés avec succès aux files d'attente qui se résolvent en files d'attente éloignées.

.NET ne prend pas en charge les listes de distribution, mais InvalidDestCount est défini lors de l'ouverture d'une file d'attente unique.

## **Constructeurs**

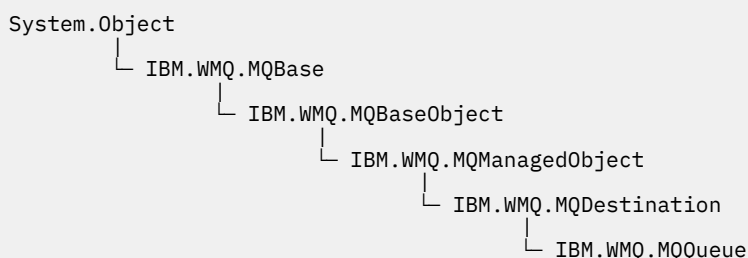
### **public MQPutMessageOptions();**

Construisez un nouvel objet MQPutMessageOptions sans option définie et un nom ResolvedQueue et un nom ResolvedQueueManagerName vide.

## **Classe MQQueue.NET**

Utilisez MQQueue pour envoyer et recevoir des messages et des attributs de requête d'une file d'attente IBM MQ . Créez un objet MQQueue à l'aide d'un constructeur ou d'une méthode MQQueueManager.AccessProcess .

## **Classe**



```
public class IBM.WMQ.MQQueue extends IBM.WMQ.MQDestination;
```

- «Propriétés», à la page 1838
- «Méthodes», à la page 1840
- «Constructeurs», à la page 1842

## Propriétés

Test de l'émission de MQException lors de l'obtention des propriétés.

```
public int ClusterWorkLoadPriority {get;}
```

Indique la priorité de la file d'attente. Ce paramètre est valide uniquement pour les files d'attente locales, distantes et alias.

```
public int ClusterWorkLoadRank {get;}
```

Indique le rang de la file d'attente. Ce paramètre est valide uniquement pour les files d'attente locales, distantes et alias.

```
public int ClusterWorkLoadUseQ {get;}
```

Indique le comportement d'une opération MQPUT lorsque la file d'attente cible comporte une instance locale et au moins une instance de cluster distant. Ce paramètre ne s'applique pas si MQPUT provient d'un canal de cluster. Ce paramètre est valide uniquement pour les files d'attente locales.

```
public DateTime CreationDateTime {get;}
```

Date et heure de création de cette file d'attente.

```
public int CurrentDepth {get;}
```

Extrait le nombre de messages actuellement dans la file d'attente. Cette valeur est incrémentée lors d'un appel d'insertion et lors de l'annulation d'un appel d'extraction. Il est décrémenté lors d'une opération get non-browse et lors de l'annulation d'un appel put.

```
public int DefinitionType {get;}
```

Mode de définition de la file d'attente. Les valeurs possibles sont les suivantes:

- MQC.MQQDT\_PREDEFINED
- MQC.MQQDT\_PERMANENT\_DYNAMIC
- MQC.MQQDT\_TEMPORARY\_DYNAMIC

```
public int InhibitGet {get; set;}
```

Contrôle si vous pouvez obtenir des messages dans cette file d'attente ou pour cette rubrique. Les valeurs possibles sont les suivantes:

- MQC.MQQA\_GET\_INHIBITED
- MQC.MQQA\_GET\_ALLOWED

```
public int InhibitPut {get; set;}
```

Contrôle si vous pouvez placer des messages dans cette file d'attente ou pour cette rubrique. Les valeurs possibles sont les suivantes:

- MQQA\_PUT\_INHIBITED
- MQQA\_PUT\_ALLOWED

```
public int MaximumDepth {get;}
```

Nombre maximal de messages pouvant exister simultanément dans la file d'attente. Une tentative d'insertion d'un message dans une file d'attente qui contient déjà ce nombre de messages échoue avec le code anomalie MQC.MQRC\_Q\_FULL.

```
public int MaximumMessageLength {get;}
```

Longueur maximale des données d'application pouvant exister dans chaque message de cette file d'attente. Une tentative d'insertion d'un message supérieur à cette valeur échoue avec le code anomalie MQC.MQRC\_MSG\_TOO\_BIG\_FOR\_Q.

```
public int NonPersistentMessageClass {get;}
```

Niveau de fiabilité des messages non persistants insérés dans cette file d'attente.

**public int OpenInputCount {get;}**

Nombre de descripteurs actuellement valides pour supprimer des messages de la file d'attente. OpenInputCount est le nombre total de descripteurs d'entrée valides connus du gestionnaire de files d'attente local, et pas seulement les descripteurs créés par l'application.

**public int OpenOutputCount {get;}**

Nombre de descripteurs actuellement valides pour l'ajout de messages à la file d'attente. OpenOutputCount est le nombre total de descripteurs de sortie valides connus du gestionnaire de files d'attente local, et pas seulement les descripteurs créés par l'application.

**public int QueueAccounting {get;}**

Indique si vous pouvez activer la collecte des informations de comptabilité pour la file d'attente.

**public int QueueMonitoring {get;}**

Indique si vous pouvez activer la surveillance de la file d'attente.

**public int QueueStatistics {get;}**

Indique si vous pouvez activer la collecte de statistiques pour la file d'attente.

**public int QueueType {get;}**

Type de cette file d'attente avec l'une des valeurs suivantes:

- MQC.MQQT\_ALIAS
- MQC.MQQT\_LOCAL
- MQC.MQQT\_REMOTE
- MQC.MQQT\_CLUSTER

**public int Shareability {get;}**

Indique si la file d'attente peut être ouverte en entrée plusieurs fois. Les valeurs possibles sont les suivantes:

- MQC.MQQA\_SHAREABLE
- MQC.MQQA\_NOT\_SHAREABLE

**public string TPIPE {get;}**

Nom TPIPE utilisé pour la communication avec OTMA à l'aide de la passerelle IBM MQ IMS .

**public int TriggerControl {get; set;}**

Indique si les messages de déclenchement sont écrits dans une file d'attente d'initialisation, afin de démarrer une application pour traiter la file d'attente. Les valeurs possibles sont les suivantes:

- MQC.MQTC\_OFF
- MQC.MQTC\_ON

**public string TriggerData {get; set;}**

Données de format libre que le gestionnaire de files d'attente insère dans le message de déclenchement. Il insère TriggerData lorsqu'un message arrivant dans cette file d'attente entraîne l'écriture d'un message de déclenchement dans la file d'attente d'initialisation. La longueur maximale autorisée de la chaîne est donnée par MQC.MQ\_TRIGGER\_DATA\_LENGTH.

**public int TriggerDepth {get; set;}**

Nombre de messages qui doivent être dans la file d'attente avant qu'un message de déclenchement ne soit écrit lorsque le type de déclencheur est défini sur MQC.MQTT\_DEPTH.

**public int TriggerMessagePriority {get; set;}**

Priorité de message sous laquelle les messages ne contribuent pas à la génération de messages de déclenchement. Autrement dit, le gestionnaire de files d'attente ignore ces messages lorsqu'il décide de générer ou non un déclencheur. La valeur zéro entraîne la contribution de tous les messages à la génération de messages de déclenchement.

**public int TriggerType {get; set;}**

Conditions dans lesquelles les messages de déclenchement sont écrits suite à l'arrivée de messages dans cette file d'attente. Les valeurs possibles sont les suivantes:

- MQC.MQTT\_NONE

- MQC.MQTT\_FIRST
- MQC.MQTT\_EVERY
- MQC.MQTT\_DEPTH

## Méthodes

```
public void Get(MQMessage message);
public void Get(MQMessage message, MQGetMessageOptions getMessageOptions);
public void Get(MQMessage message, MQGetMessageOptions getMessageOptions, int
MaxMsgSize);
```

Emet MQException.

Extrait un message d'une file d'attente.

Si l'extraction échoue, l'objet MQMessage n'est pas modifié. Si elle aboutit, le descripteur de message et les parties de données de message du MQMessage sont remplacés par le descripteur de message et les données de message du message entrant.

Tous les appels à IBM MQ à partir d'un MQQueueManager particulier sont synchrones. Par conséquent, si vous effectuez une opération get avec attente, toutes les autres unités d'exécution utilisant le même MQQueueManager ne peuvent pas effectuer d'autres appels IBM MQ jusqu'à ce que l'appel Get soit effectué. Si vous avez besoin de plusieurs unités d'exécution pour accéder à IBM MQ simultanément, chaque unité d'exécution doit créer son propre objet MQQueueManager .

### Message

Contient le descripteur de message et les données de message renvoyées. Certaines des zones du descripteur de message sont des paramètres d'entrée. Il est important de s'assurer que les paramètres d'entrée MessageId et CorrelationId sont définis comme requis.

Un client reconnectable renvoie le code anomalie MQRC\_BACKED\_OUT après une reconnexion réussie, pour les messages reçus sous MQGM\_SYNCPOINT.

### Options de getMessage

Options contrôlant l'action de la commande get.

L'utilisation de l'option MQC.MQGM\_CONVERT peut générer une exception avec le code anomalie MQC.MQRC\_CONVERTED\_STRING\_TOO\_BIG lors de la conversion de codes de caractères codés sur un octet en codes codés sur deux octets. Dans ce cas, le message est copié dans la mémoire tampon sans conversion.

Si *getMessageOptions* n'est pas spécifié, l'option de message utilisée est MQGM\_NOWAIT.

Si vous utilisez l'option MQGM\_LOGICAL\_ORDER dans un client reconnectable, le code anomalie MQRC\_RECONNECT\_INCOMPATIBLE est renvoyé.

### Taille MaxMsg

Message le plus volumineux que cet objet message doit recevoir. Si le message de la file d'attente est supérieur à cette taille, l'un des deux événements suivants se produit:

- Si l'indicateur MQGM\_ACCEPT\_TRUNCATED\_MSG est défini dans l'objet MQGetMessageOptions , le message est rempli avec autant de données de message que possible. Une exception est émise avec le code achèvement MQCC\_WARNING et le code anomalie MQRC\_TRUNCATED\_MSG\_ACCEPTED .
- Si l'indicateur MQGM\_ACCEPT\_TRUNCATED\_MSG n'est pas défini, le message reste dans la file d'attente. Une exception est émise avec le code achèvement MQCC\_WARNING et le code anomalie MQRC\_TRUNCATED\_MSG\_FAILED .

Si *MaxMsgSize* n'est pas spécifié, l'intégralité du message est extraite.

```
public void Put(MQMessage message);
public void Put(MQMessage message, MQPutMessageOptions putMessageOptions);
```

Emet MQException.



Insère un message dans une file d'attente.

Les modifications apportées à l'objet `MQMessage` après l'appel d'insertion n'affectent pas le message réel dans la file d'attente IBM MQ ou dans la rubrique de publication.

`Put` met à jour les propriétés `MessageId` et `CorrelationId` de l'objet `MQMessage` et n'efface pas les données de message. Les appels `Put` ou `Get` supplémentaires font référence aux informations mises à jour dans l'objet `MQMessage`. Par exemple, dans le fragment de code suivant, le premier message contient a et le second ab.

```
msg.WriteString("a");
q.Put(msg, pmo);
msg.WriteString("b");
q.Put(msg, pmo);
```

### Message

Objet `MQMessage` contenant les données du descripteur de message et le message à envoyer. Le descripteur de message peut être modifié à la suite de cette méthode. Les valeurs du descripteur de message immédiatement après la fin de cette méthode sont les valeurs qui ont été placées dans la file d'attente ou publiées dans la rubrique.

Les codes anomalie suivants sont renvoyés à un client reconnectable:

- `MQRC_CALL_INTERRUPTED` si la connexion est interrompue lors de l'exécution d'un appel `Put` sur un message persistant et que la reconnexion a abouti.
- `MQRC_NONE` si la connexion aboutit lors de l'exécution d'un appel `Put` sur un message non persistant (voir [Application Recovery](#)).

### Options de `putMessage`

Options contrôlant l'action de l'insertion.

Si `putMessageOptions` n'est pas spécifié, l'instance par défaut de `MQPutMessageOptions` est utilisée.

Si vous utilisez l'option `MQPMO_LOGICAL_ORDER` dans un client reconnectable, le code anomalie `MQRC_RECONNECT_INCOMPATIBLE` est renvoyé.

**Remarque :** Pour plus de simplicité et de performances, si vous souhaitez placer un seul message dans une file d'attente, utilisez l'objet `MQQueueManager.Put`. Vous devez disposer d'un objet `MQQueue` pour cela.

```
public void PutForwardMessage(MQMessage message);  
public void PutForwardMessage(MQMessage message, MQPutMessageOptions  
putMessageOptions);
```

Emissions `MQException`

Insérez un message en cours de réacheminement dans la file d'attente, où `message` est le message d'origine.

### Message

Objet `MQMessage` contenant les données du descripteur de message et le message à envoyer. Le descripteur de message peut être modifié à la suite de cette méthode. Les valeurs du descripteur de message immédiatement après la fin de cette méthode sont les valeurs qui ont été placées dans la file d'attente ou publiées dans la rubrique.

Les codes anomalie suivants sont renvoyés à un client reconnectable:

- `MQRC_CALL_INTERRUPTED` si la connexion est interrompue lors de l'exécution d'un appel `Put` sur un message persistant et que la reconnexion a abouti.
- `MQRC_NONE` si la connexion aboutit lors de l'exécution d'un appel `Put` sur un message non persistant (voir [Application Recovery](#)).

### Options de `putMessage`

Options contrôlant l'action de l'insertion.

Si *putMessageOptions* n'est pas spécifié, l'instance par défaut de `MQPutMessageOptions` est utilisée.

Si vous utilisez l'option `MQPMO_LOGICAL_ORDER` dans un client reconnectable, le code anomalie `MQRC_RECONNECT_INCOMPATIBLE` est renvoyé.

```
public void PutReplyMessage(MQMessage message)  
public void PutReplyMessage(MQMessage message, MQPutMessageOptions  
putMessageOptions)
```

Emet `MQException`.

Placez un message de réponse dans la file d'attente, où *message* est le message d'origine.

#### **Message**

Contient le descripteur de message et les données de message renvoyées. Certaines des zones du descripteur de message sont des paramètres d'entrée. Il est important de s'assurer que les paramètres d'entrée `MessageId` et `CorrelationId` sont définis comme requis.

Un client reconnectable renvoie le code anomalie `MQRC_BACKED_OUT` après une reconnexion réussie, pour les messages reçus sous `MQGM_SYNCPOINT`.

#### **Options de putMessage**

Options contrôlant l'action de l'insertion.

Si *putMessageOptions* n'est pas spécifié, l'instance par défaut de `MQPutMessageOptions` est utilisée.

Si vous utilisez l'option `MQPMO_LOGICAL_ORDER` dans un client reconnectable, le code anomalie `MQRC_RECONNECT_INCOMPATIBLE` est renvoyé.

```
public void PutReportMessage(MQMessage message)  
public void PutReportMessage(MQMessage message, MQPutMessageOptions  
putMessageOptions)
```

Emet `MQException`.

Placez un message de rapport dans la file d'attente, où *message* est le message d'origine.

#### **Message**

Contient le descripteur de message et les données de message renvoyées. Certaines des zones du descripteur de message sont des paramètres d'entrée. Il est important de s'assurer que les paramètres d'entrée `MessageId` et `CorrelationId` sont définis comme requis.

Un client reconnectable renvoie le code anomalie `MQRC_BACKED_OUT` après une reconnexion réussie, pour les messages reçus sous `MQGM_SYNCPOINT`.

#### **Options de putMessage**

Options contrôlant l'action de l'insertion.

Si *putMessageOptions* n'est pas spécifié, l'instance par défaut de `MQPutMessageOptions` est utilisée.

Si vous utilisez l'option `MQPMO_LOGICAL_ORDER` dans un client reconnectable, le code anomalie `MQRC_RECONNECT_INCOMPATIBLE` est renvoyé.

## **Constructeurs**

```
public MQQueue MQQueueManager.AccessQueue(string queueName, int openOptions);  
public MQQueue MQQueueManager.AccessQueue(string queueName, int openOptions,  
string queueManagerName, string dynamicQueueName, string alternateUserId);
```

Emet `MQException`.

Accède à une file d'attente sur ce gestionnaire de files d'attente.

Vous pouvez extraire ou parcourir des messages, insérer des messages, demander des informations sur les attributs de la file d'attente ou définir les attributs de la file d'attente. Si la file d'attente

nommée est une file d'attente modèle, une file d'attente locale dynamique est créée. Interrogez l'attribut name de l'objet MQQueue résultant pour trouver le nom de la file d'attente dynamique.

#### **queueName**

Nom de la file d'attente à ouvrir.

#### **openOptions**

Options qui contrôlent l'ouverture de la file d'attente.

##### **MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY**

Valider avec l'ID utilisateur spécifié.

##### **MQC.MQOO\_BIND\_AS\_QDEF**

Utilisez la liaison par défaut pour la file d'attente.

##### **MQC.MQOO\_BIND\_NOT\_FIXED**

Ne vous liez pas à une destination spécifique.

##### **MQC.MQOO\_BIND\_ON\_OPEN**

Descripteur de liaison à la destination lors de l'ouverture de la file d'attente.

##### **MQC.MQOO\_BROWSE**

Ouvrir pour parcourir le message.

##### **MQC.MQOO\_FAIL\_IF QUIESCING**

Echec si le gestionnaire de files d'attente est au repos.

##### **MQC.MQOO\_INPUT\_AS\_Q\_DEF**

Ouvrir pour obtenir les messages à l'aide de la valeur par défaut définie par la file d'attente.

##### **MQC.MQOO\_INPUT\_SHARED**

Ouvert pour obtenir les messages avec un accès partagé.

##### **MQC.MQOO\_INPUT\_EXCLUSIVE**

Ouvrir pour obtenir des messages avec un accès exclusif.

##### **MQC.MQOO\_INQUIRE**

Ouvert pour interrogation-obligatoire si vous souhaitez interroger les propriétés.

##### **MQC.MQOO\_OUTPUT**

Ouvrir pour insérer des messages.

##### **MQC.MQOO\_PASS\_ALL\_CONTEXT**

Autoriser la transmission de tous les contextes.

##### **MQC.MQOO\_PASS\_IDENTITY\_CONTEXT**

Autoriser la transmission du contexte d'identité.

##### **MQC.MQOO\_SAVE\_ALL\_CONTEXT**

Sauvegarder le contexte lors de l'extraction du message.

##### **MQC.MQOO\_SET**

Ouvert pour définir des attributs-requis si vous souhaitez définir des propriétés.

##### **MQC.MQOO\_SET\_ALL\_CONTEXT**

Permet de définir tous les contextes.

##### **MQC.MQOO\_SET\_IDENTITY\_CONTEXT**

Permet de définir le contexte d'identité.

#### **QueueManagerName**

Nom du gestionnaire de files d'attente sur lequel la file d'attente est définie. Un nom entièrement vide ou null indique le gestionnaire de files d'attente auquel l'objet MQQueueManager est connecté.

#### **dynamicQueueNom**

*dynamicQueueName* est ignoré sauf si *queueName* spécifie le nom d'une file d'attente modèle. Si tel est le cas, *dynamicQueueName* indique le nom de la file d'attente dynamique à créer. Un nom vide ou nul n'est pas valide si *queueName* spécifie le nom d'une file d'attente modèle. Si le dernier caractère non blanc du nom est un astérisque, \*, le gestionnaire de files d'attente remplace

l'astérisque par une chaîne de caractères. Les caractères garantissent que le nom généré pour la file d'attente est unique sur ce gestionnaire de files d'attente.

#### **AlternateUserId**

Si MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY est spécifié dans le paramètre openOptions , *alternateUserId* indique l'autre identificateur utilisateur utilisé pour vérifier l'autorisation d'ouverture. Si MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY n'est pas spécifié, *alternateUserId* peut être laissé vide ou null.

```
public MQQueue(MQQueueManager queueManager, string queueName, int openOptions,  
string queueManagerName, string dynamicQueueName, string alternateUserId);
```

Emet MQException.

Accède à une file d'attente sur queueManager.

Vous pouvez extraire ou parcourir des messages, insérer des messages, demander des informations sur les attributs de la file d'attente ou définir les attributs de la file d'attente. Si la file d'attente nommée est une file d'attente modèle, une file d'attente locale dynamique est créée. Interrogez l'attribut name de l'objet MQQueue résultant pour trouver le nom de la file d'attente dynamique.

#### **queueManager**

Gestionnaire de files d'attente sur lequel accéder à la file d'attente.

#### **queueName**

Nom de la file d'attente à ouvrir.

#### **openOptions**

Options qui contrôlent l'ouverture de la file d'attente.

#### **MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY**

Valider avec l'ID utilisateur spécifié.

#### **MQC.MQOO\_BIND\_AS\_QDEF**

Utilisez la liaison par défaut pour la file d'attente.

#### **MQC.MQOO\_BIND\_NOT\_FIXED**

Ne vous liez pas à une destination spécifique.

#### **MQC.MQOO\_BIND\_ON\_OPEN**

Descripteur de liaison à la destination lors de l'ouverture de la file d'attente.

#### **MQC.MQOO\_BROWSE**

Ouvrir pour parcourir le message.

#### **MQC.MQOO\_FAIL\_IF QUIESCING**

Echec si le gestionnaire de files d'attente est au repos.

#### **MQC.MQOO\_INPUT\_AS\_Q\_DEF**

Ouvrir pour obtenir les messages à l'aide de la valeur par défaut définie par la file d'attente.

#### **MQC.MQOO\_INPUT\_SHARED**

Ouvert pour obtenir les messages avec un accès partagé.

#### **MQC.MQOO\_INPUT\_EXCLUSIVE**

Ouvrir pour obtenir des messages avec un accès exclusif.

#### **MQC.MQOO\_INQUIRE**

Ouvert pour interrogation-obligatoire si vous souhaitez interroger les propriétés.

#### **MQC.MQOO\_OUTPUT**

Ouvrir pour insérer des messages.

#### **MQC.MQOO\_PASS\_ALL\_CONTEXT**

Autoriser la transmission de tous les contextes.

#### **MQC.MQOO\_PASS\_IDENTITY\_CONTEXT**

Autoriser la transmission du contexte d'identité.

#### **MQC.MQOO\_SAVE\_ALL\_CONTEXT**

Sauvegarder le contexte lors de l'extraction du message.

### **MQC.MQOO\_SET**

Ouvert pour définir des attributs-requis si vous souhaitez définir des propriétés.

### **MQC.MQOO\_SET\_ALL\_CONTEXT**

Permet de définir tous les contextes.

### **MQC.MQOO\_SET\_IDENTITY\_CONTEXT**

Permet de définir le contexte d'identité.

### **QueueManagerName**

Nom du gestionnaire de files d'attente sur lequel la file d'attente est définie. Un nom entièrement vide ou null indique le gestionnaire de files d'attente auquel l'objet MQQueueManager est connecté.

### **dynamicQueueNom**

*dynamicQueueName* est ignoré sauf si *queueName* spécifie le nom d'une file d'attente modèle. Si tel est le cas, *dynamicQueueName* indique le nom de la file d'attente dynamique à créer. Un nom vide ou nul n'est pas valide si *queueName* spécifie le nom d'une file d'attente modèle. Si le dernier caractère non blanc du nom est un astérisque, \*, le gestionnaire de files d'attente remplace l'astérisque par une chaîne de caractères. Les caractères garantissent que le nom généré pour la file d'attente est unique sur ce gestionnaire de files d'attente.

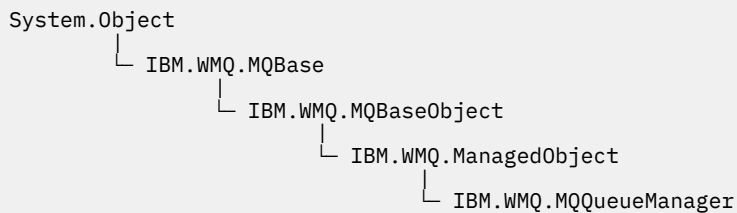
### **AlternateUserId**

Si MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY est spécifié dans le paramètre *openOptions*, *alternateUserId* indique l'autre identificateur utilisateur utilisé pour vérifier l'autorisation d'ouverture. Si MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY n'est pas spécifié, *alternateUserId* peut être laissé vide ou null.

## **Classe MQQueueManager.NET**

Utilisez MQQueueManager pour vous connecter à un gestionnaire de files d'attente et accéder aux objets de gestionnaire de files d'attente. Il contrôle également les transactions. Le constructeur MQQueueManager crée une connexion client ou serveur.

### **Classe**



```
public class IBM.WMQ.MQQueueManager extends IBM.WMQ.MQManagedObject;
```

- «Propriétés», à la page 1845
- «Méthodes», à la page 1849
- «Constructeurs», à la page 1855

### **Propriétés**

Test de l'émission de MQException lors de l'obtention des propriétés.

```
public int AccountingConnOverride {get;}
```

Indique si les applications peuvent remplacer les valeurs de comptabilité et de comptabilité de file d'attente MQI .

```
public int AccountingInterval {get;}
```

Durée avant l'écriture des enregistrements comptables intermédiaires (en secondes).

**public int ActivityRecording {get;}**  
Ctrl la génération des rapports d'activité.

**public int AdoptNewMCACheck {get;}**  
Indique quels éléments sont vérifiés pour déterminer si l'agent MCA est adopté lorsqu'un nouveau canal entrant est détecté. Pour être adopté, le nom de l'agent MCA doit correspondre à celui d'un agent MCA actif.

**public int AdoptNewMCAInterval {get;}**  
Durée, en secondes, pendant laquelle le nouveau canal attend l'arrêt du canal orphelin.

**public int AdoptNewMCAType {get;}**  
Indique si une instance MCA orpheline doit être adoptée (redémarrée) lorsqu'une nouvelle demande de canal entrant correspondant à la valeur MCACheck AdoptNewest détectée.

**public int BridgeEvent {get;}**  
Indique si les événements de pont IMS sont générés.

**public int ChannelEvent {get;}**  
Indique si des événements de canal sont générés.

**public int ChannelInitiatorControl {get;}**  
Indique si l'initiateur de canal démarre automatiquement au démarrage du gestionnaire de files d'attente.

**public int ChannelInitiatorAdapters {get;}**  
Nombre de sous-tâches de l'adaptateur pour le traitement des appels IBM MQ .

**public int ChannelInitiatorDispatchers {get;}**  
Nombre de répartiteurs à utiliser pour l'initiateur de canal.

**public int ChannelInitiatorTraceAutoStart {get;}**  
Indique si la trace de l'initiateur de canal démarre automatiquement.

**public int ChannelInitiatorTraceTableSize {get;}**  
Taille, en mégaoctets, de l'espace de données de trace d'un initiateur de canal.

**public int ChannelMonitoring {get;}**  
Indique si la surveillance des canaux est utilisée.

**public int ChannelStatistics {get;}**  
Ctrl la collecte des données de statistiques pour les canaux.

**public int CharacterSet {get;}**  
Renvoie l'ID de jeu de caractères codés (CCSID) du gestionnaire de files d'attente. CharacterSet est utilisé par le gestionnaire de files d'attente pour toutes les zones de chaîne de caractères de l'interface de programme d'application.

**public int ClusterSenderMonitoring {get;}**  
Contrôle la collecte des données de surveillance en ligne pour les canaux émetteurs de cluster définis automatiquement.

**public int ClusterSenderStatistics {get;}**  
Contrôle la collecte des données statistiques pour les canaux émetteurs de cluster définis automatiquement.

**public int ClusterWorkLoadMRU {get;}**  
Nombre maximal de canaux de cluster sortants.

**public int ClusterWorkLoadUseQ {get;}**  
La valeur par défaut de la propriété MQQueue , ClusterWorkLoadUseQ, si elle spécifie la valeur QMGR.

**public int CommandEvent {get;}**  
Définit si des événements Commande sont générés.

**public string CommandInputQueueName {get;}**  
Renvoie le nom de la file d'attente d'entrée de commandes définie sur le gestionnaire de files d'attente. Les applications peuvent envoyer des commandes à cette file d'attente, si elles y sont autorisées.

**public int CommandLevel {get;}**

Indique le niveau de fonction du gestionnaire de files d'attente. L'ensemble des fonctions qui correspondent à un niveau de fonction particulier dépend de la plateforme. Sur une plateforme particulière, vous pouvez vous appuyer sur chaque gestionnaire de files d'attente prenant en charge les fonctions au niveau fonctionnel le plus bas commun à tous les gestionnaires de files d'attente.

**public int CommandLevel {get;}**

Indique si le serveur de commandes démarre automatiquement au démarrage du gestionnaire de files d'attente.

**public string DNSGroup {get;}**

Obsolète.

**public int DNSWLM {get;}**

Obsolète.

**public int IPAddressVersion {get;}**

Protocole IP (IPv4 ou IPv6) à utiliser pour une connexion de canal.

**public boolean IsConnected {get;}**

Renvoie la valeur de `isConnected`.

Si la valeur est true, une connexion au gestionnaire de files d'attente a été établie et n'est pas rompue. Tous les appels à `IsConnected` ne tentent pas activement d'atteindre le gestionnaire de files d'attente. Il est donc possible que la connectivité physique soit rompée, mais `IsConnected` peut toujours renvoyer la valeur true. L'état `IsConnected` est uniquement mis à jour lorsque l'activité, par exemple, l'insertion d'un message, l'obtention d'un message, est effectuée sur le gestionnaire de files d'attente.

Si la valeur est false, une connexion au gestionnaire de files d'attente n'a pas été établie, a été interrompue ou a été déconnectée.

**public int KeepAlive {get;}**

Indique si la fonction TCP KEEPALIVE doit être utilisée pour vérifier que l'autre extrémité de la connexion est toujours disponible. S'il n'est pas disponible, le canal est fermé.

**public int ListenerTimer {get;}**

Intervalle, en secondes, entre les tentatives de redémarrage du programme d'écoute effectuées par IBM MQ après un échec APPC ou TCP/IP.

**public int LoggerEvent {get;}**

Indique si les événements du consignateur sont générés.

**public string LU62ARMSuffix {get;}**

Suffixe du membre APPCPM de SYS1.PARMLIB. Ce suffixe désigne la LUADD de cet initiateur de canal. Lorsque le gestionnaire de redémarrage automatique (ARM) redémarre l'initiateur de canal, la commande z/OS SET APPC=xx est émise.

**public string LUGroupName {get; z/os}**

Nom d'unité logique générique à utiliser par le programme d'écoute d'unité logique 6.2 qui gère les transmissions entrantes pour le groupe de partage de files d'attente.

**public string LUName {get;}**

Nom de l'unité logique à utiliser pour les transmissions LU 6.2 sortantes.

**public int MaximumActiveChannels {get;}**

Le nombre maximal de canaux pouvant être actifs en même temps.

**public int MaximumCurrentChannels {get;}**

Nombre maximal de canaux pouvant être à jour à tout moment (y compris les canaux de connexion serveur avec des clients connectés).

**public int MaximumLU62Channels {get;}**

Nombre maximal de canaux pouvant être en cours, ou de clients pouvant être connectés, qui utilisent le protocole de transmission LU 6.2 .

**public int MaximumMessageLength {get;}**

Renvoie la longueur maximale d'un message (en octets) pouvant être traité par le gestionnaire de files d'attente. Aucune file d'attente ne peut être définie avec une longueur de message maximale supérieure à `MaximumMessageLength`.

**public int MaximumPriority {get;}**

Renvoie la priorité maximale des messages prise en charge par le gestionnaire de files d'attente. Les priorités sont comprises entre zéro (valeur la plus faible) et cette valeur. Emet `MQException` si vous appelez cette méthode après la déconnexion du gestionnaire de files d'attente.

**public int MaximumTCPChannels {get;}**

Nombre maximal de canaux pouvant être en cours, ou de clients pouvant être connectés, qui utilisent le protocole de transmission TCP/IP.

**public int MQIAccounting {get;}**

Ctrl la collecte des informations comptables pour les données MQI.

**public int MQIStatistics {get;}**

Ctrl la collecte d'informations de contrôle de statistiques pour le gestionnaire de files d'attente.

**public int OutboundPortMax {get;}**

Valeur maximale dans la plage de numéros de port à utiliser lors de la liaison de canaux sortants.

**public int OutboundPortMin {get;}**

Valeur minimale dans la plage de numéros de port à utiliser lors de la liaison de canaux sortants.

**public int QueueAccounting {get;}**

Indique si les données de comptabilité de classe 3 (comptabilité de niveau unité d'exécution et de niveau file d'attente) doivent être utilisées pour toutes les files d'attente.

**public int QueueMonitoring {get;}**

Ctrl la collecte des données de surveillance en ligne pour les files d'attente.

**public int QueueStatistics {get;}**

Ctrl la collecte des données de statistiques pour les files d'attente.

**public int ReceiveTimeout {get;}**

Durée pendant laquelle un canal TCP/IP attend de recevoir des données, y compris des signaux de présence, de la part de son partenaire avant de revenir à l'état inactif.

**public int ReceiveTimeoutMin {get;}**

Durée minimale pendant laquelle un canal TCP/IP attend de recevoir des données, y compris des signaux de présence, de la part de son partenaire avant de revenir à l'état inactif.

**public int ReceiveTimeoutType {get;}**

Qualificateur à appliquer à la valeur dans `ReceiveTimeout`.

**public int SharedQueueQueueManagerName {get;}**

Indique comment distribuer des messages dans une file d'attente partagée. Si l'insertion spécifie un gestionnaire de files d'attente différent du même groupe de partage de files d'attente que le gestionnaire de files d'attente cible, le message est distribué de deux manières:

**MQC.MQSQQM\_USE**

Les messages sont distribués au gestionnaire de files d'attente d'objets avant d'être placés dans la file d'attente partagée.

**MQCMQSQQM\_IGNORE**

Les messages sont placés directement dans la file d'attente partagée.

**public int SSLEvent {get;}**

Indique si des événements TLS sont générés.

**public int SSLFips {get;}**

Indique si seuls les algorithmes certifiés FIPS doivent être utilisés si la cryptographie est effectuée dans IBM MQ, plutôt que dans du matériel de cryptographie.

**public int SSLKeyResetCount {get;}**

Indique le nombre d'octets non chiffrés envoyés et reçus dans une conversation TLS avant la renégociation de la clé secrète.



**public int ClusterSenderStatistics {get;}**

Indique l'intervalle, en minutes, entre les collectes de statistiques consécutives.

**public int SyncpointAvailability {get;}**

Indique si le gestionnaire de files d'attente prend en charge les unités de travail et les points de synchronisation avec les méthodes `MQQueue.get` et `MQQueue.put`.

**public string TCPName {get;}**

Nom du système TCP/IP unique ou par défaut à utiliser, en fonction de la valeur de `TCPStackType`.

**public int TCPStackType {get;}**

Indique si l'initiateur de canal utilise uniquement l'espace adresse TCP/IP spécifié dans `TCPName`. L'initiateur de canal peut également se connecter à n'importe quelle adresse TCP/IP.

**public int TraceRouteRecording {get;}**

Contrôle l'enregistrement des informations de trace de route.

## Méthodes

**public MQProcess AccessProcess(string processName, int openOptions);**

**public MQProcess AccessProcess(string processName, int openOptions, string queueManagerName, string alternateUserId);**

Emet `MQException`.

Accédez à un processus IBM MQ sur ce gestionnaire de files d'attente pour consulter les attributs de processus.

### **processName**

Nom du processus à ouvrir.

### **openOptions**

Options qui contrôlent l'ouverture du processus. Les options valides qui peuvent être ajoutées ou combinées à l'aide d'un OR bit à bit sont les suivantes:

- `MQC.MQOO_FAIL_IF QUIESCING`
- `MQC.MQOO_INQUIRE`
- `MQC.MQOO_SET`
- `MQC.MQOO_ALTERNATE_USER_AUTHORITY`

### **QueueManagerName**

Nom du gestionnaire de files d'attente sur lequel le processus est défini. Vous pouvez laisser un nom de gestionnaire de files d'attente vide ou null si le gestionnaire de files d'attente est identique à celui auquel le processus accède.

### **AlternateUserId**

Si `MQC.MQOO_ALTERNATE_USER_AUTHORITY` est spécifié dans le paramètre **openOptions**, *alternateUserId* indique l'autre ID utilisateur utilisé pour vérifier l'autorisation de l'action. Si `MQOO_ALTERNATE_USER_AUTHORITY` n'est pas spécifié, *alternateUserId* peut être vide ou null.

Les droits utilisateur par défaut sont utilisés pour la connexion au gestionnaire de files d'attente si `MQC.MQOO_ALTERNATE_USER_AUTHORITY` n'est pas spécifié.

**public MQQueue AccessQueue(string queueName, int openOptions);**

**public MQQueue AccessQueue(string queueName, int openOptions, string queueManagerName, string dynamicQueueName, string alternateUserId);**

Emet `MQException`.

Accède à une file d'attente sur ce gestionnaire de files d'attente.

Vous pouvez extraire ou parcourir des messages, insérer des messages, demander des informations sur les attributs de la file d'attente ou définir les attributs de la file d'attente. Si la file d'attente

nommée est une file d'attente modèle, une file d'attente locale dynamique est créée. Interrogez l'attribut name de l'objet MQQueue résultant pour trouver le nom de la file d'attente dynamique.

**queueName**

Nom de la file d'attente à ouvrir.

**openOptions**

Options qui contrôlent l'ouverture de la file d'attente.

**MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY**

Valider avec l'ID utilisateur spécifié.

**MQC.MQOO\_BIND\_AS\_QDEF**

Utilisez la liaison par défaut pour la file d'attente.

**MQC.MQOO\_BIND\_NOT\_FIXED**

Ne vous liez pas à une destination spécifique.

**MQC.MQOO\_BIND\_ON\_OPEN**

Descripteur de liaison à la destination lors de l'ouverture de la file d'attente.

**MQC.MQOO\_BROWSE**

Ouvrir pour parcourir le message.

**MQC.MQOO\_FAIL\_IF QUIESCING**

Echec si le gestionnaire de files d'attente est au repos.

**MQC.MQOO\_INPUT\_AS\_Q\_DEF**

Ouvrir pour obtenir les messages à l'aide de la valeur par défaut définie par la file d'attente.

**MQC.MQOO\_INPUT\_SHARED**

Ouvert pour obtenir les messages avec un accès partagé.

**MQC.MQOO\_INPUT\_EXCLUSIVE**

Ouvrir pour obtenir des messages avec un accès exclusif.

**MQC.MQOO\_INQUIRE**

Ouvert pour interrogation-obligatoire si vous souhaitez interroger les propriétés.

**MQC.MQOO\_OUTPUT**

Ouvrir pour insérer des messages.

**MQC.MQOO\_PASS\_ALL\_CONTEXT**

Autoriser la transmission de tous les contextes.

**MQC.MQOO\_PASS\_IDENTITY\_CONTEXT**

Autoriser la transmission du contexte d'identité.

**MQC.MQOO\_SAVE\_ALL\_CONTEXT**

Sauvegarder le contexte lors de l'extraction du message.

**MQC.MQOO\_SET**

Ouvert pour définir des attributs-requis si vous souhaitez définir des propriétés.

**MQC.MQOO\_SET\_ALL\_CONTEXT**

Permet de définir tous les contextes.

**MQC.MQOO\_SET\_IDENTITY\_CONTEXT**

Permet de définir le contexte d'identité.

**QueueManagerName**

Nom du gestionnaire de files d'attente sur lequel la file d'attente est définie. Un nom entièrement vide ou null indique le gestionnaire de files d'attente auquel l'objet MQQueueManager est connecté.

**dynamicQueueNom**

*dynamicQueueName* est ignoré sauf si *queueName* spécifie le nom d'une file d'attente modèle. Si tel est le cas, *dynamicQueueName* indique le nom de la file d'attente dynamique à créer. Un nom vide ou nul n'est pas valide si *queueName* spécifie le nom d'une file d'attente modèle. Si le dernier caractère non blanc du nom est un astérisque, \*, le gestionnaire de files d'attente remplace

l'astérisque par une chaîne de caractères. Les caractères garantissent que le nom généré pour la file d'attente est unique sur ce gestionnaire de files d'attente.

#### **AlternateUserId**

Si MQC.MQ00\_ALTERNATE\_USER\_AUTHORITY est spécifié dans le paramètre openOptions , *alternateUserId* indique l'autre identificateur utilisateur utilisé pour vérifier l'autorisation d'ouverture. Si MQC.MQ00\_ALTERNATE\_USER\_AUTHORITY n'est pas spécifié, *alternateUserId* peut être laissé vide ou null.

```
public MQTopic AccessTopic( MQDestination destination, string topicName, string
topicObject, int options);
public MQTopic AccessTopic( MQDestination destination, string topicName, string
topicObject, int options, string alternateUserId);
public MQTopic AccessTopic( MQDestination destination, string topicName, string
topicObject, int options, string alternateUserId, string subscriptionName);
public MQTopic AccessTopic( MQDestination destination, string topicName, string
topicObject, int options, string alternateUserId, string subscriptionName,
System.Collections.Hashtable properties);
public MQTopic AccessTopic(string topicName, string topicObject, int openAs,
int options);
public MQTopic AccessTopic(string topicName, string topicObject, int openAs,
int options, string alternateUserId);
public MQTopic AccessTopic(string topicName, string topicObject, int options,
string alternateUserId, string subscriptionName);
public MQTopic AccessTopic(string topicName, string topicObject, int options,
string alternateUserId, string subscriptionName, System.Collections.Hashtable
properties);
```

Accédez à une rubrique de ce gestionnaire de files d'attente.

Les objets MQTopic sont étroitement liés aux objets de rubrique d'administration, qui sont parfois appelés objets de rubrique. En entrée, *topicObject* pointe vers un objet de rubrique d'administration. Le constructeur MQTopic obtient une chaîne de rubrique de l'objet de rubrique et la combine avec *topicName* pour créer un nom de rubrique. *topicObject* ou *topicName* peut être null. Le nom de rubrique est mis en correspondance avec l'arborescence de rubriques et le nom de l'objet de rubrique d'administration correspondant le plus proche est renvoyé dans *topicObject*.

Les rubriques associées à l'objet MQTopic sont le résultat de la combinaison de deux chaînes de rubrique. La première chaîne de rubrique est définie par l'objet de rubrique d'administration identifié par *topicObject*. La deuxième chaîne de rubrique est *topicString*. La chaîne de rubrique résultante associée à l'objet MQTopic peut identifier plusieurs rubriques en incluant des caractères génériques.

Selon que la rubrique est ouverte pour la publication ou l'abonnement, vous pouvez utiliser les méthodes MQTopic .Put pour la publication sur des rubriques ou les méthodes MQTopic .Get pour la réception de publications sur des rubriques. Si vous souhaitez publier et vous abonner à la même rubrique, vous devez accéder à la rubrique deux fois, une fois pour la publication et une fois pour l'abonnement.

Si vous créez un objet MQTopic pour l'abonnement, sans fournir d'objet MQDestination , un abonnement géré est supposé. Si vous transmettez une file d'attente en tant qu'objet MQDestination , un abonnement non géré est supposé. Vous devez vous assurer que les options d'abonnement que vous définissez sont cohérentes avec l'abonnement géré ou non géré.

#### **destination**

*destination* est une instance MQQueue . En fournissant *destination*, MQTopic est ouvert en tant qu'abonnement non géré. Les publications de la rubrique sont distribuées dans la file d'attente accessible en tant que *destination*.

#### **topicName**

Chaîne de rubrique correspondant à la deuxième partie du nom de la rubrique. *topicName* est concaténé avec la chaîne de rubrique définie dans l'objet de rubrique d'administration

*topicObject*. Vous pouvez définir *topicName* sur null, auquel cas le nom de rubrique est défini par la chaîne de rubrique dans *topicObject*.

### **topicObject**

En entrée, *topicObject* est le nom de l'objet de rubrique qui contient la chaîne de rubrique qui forme la première partie du nom de rubrique. La chaîne de rubrique dans *topicObject* est concaténée avec *topicName*. Les règles de construction des chaînes de rubrique sont définies dans [Combinaison de chaînes de rubrique](#).

Dans la sortie, *topicObject* contient le nom de l'objet de rubrique d'administration qui correspond le plus à la rubrique identifiée par la chaîne de rubrique dans l'arborescence de rubriques.

### **openAs**

Accédez à la rubrique pour la publication ou l'abonnement. Le paramètre ne peut contenir qu'une seule des options suivantes:

- MQC.MQTOPIC\_OPEN\_AS\_SUBSCRIPTION
- MQC.MQTOPIC\_OPEN\_AS\_PUBLICATION

### **options**

Combinez les options qui contrôlent l'ouverture de la rubrique pour la publication ou l'abonnement. Utilisez les constantes MQC.MQSO\_\* pour accéder à une rubrique pour l'abonnement et les constantes MQC.MQOO\_\* pour accéder à une rubrique pour la publication.

Si plusieurs options sont requises, ajoutez les valeurs ensemble ou combinez les valeurs d'option à l'aide de l'opérateur OR bit à bit.

### **AlternateUserid**

Indiquez l'ID utilisateur de remplacement utilisé pour vérifier l'autorisation requise pour terminer l'opération. Vous devez spécifier *alternateUserId*, si MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY ou MQC.MQSO\_ALTERNATE\_USER\_AUTHORITY est défini dans le paramètre d'options.

### **subscriptionName**

*subscriptionName* est obligatoire si les options MQC.MQSO\_DURABLE ou MQC.MQSO\_ALTER sont fournies. Dans les deux cas, MQTopic est implicitement ouvert pour l'abonnement. Une exception est émise si MQC.MQSO\_DURABLE est défini et que l'abonnement existe, ou si MQC.MQSO\_ALTER est défini, et que l'abonnement n'existe pas.

### **propriétés**

Définissez les propriétés d'abonnement spéciales répertoriées à l'aide d'une table de hachage. Les entrées spécifiées dans la table de hachage sont mises à jour avec les valeurs de sortie. Les entrées ne sont pas ajoutées à la table de hachage pour signaler les valeurs de sortie.

- MQC.MQSUB\_PROP\_ALTERNATE\_SECURITY\_ID
- MQC.MQSUB\_PROP\_SUBSCRIPTION\_EXPIRY
- MQC.MQSUB\_PROP\_SUBSCRIPTION\_USER\_DATA
- MQC.MQSUB\_PROP\_SUBSCRIPTION\_CORRELATION\_ID
- MQC.MQSUB\_PROP\_PUBLICATION\_PRIORITY
- MQC.MQSUB\_PROP\_PUBLICATION\_ACCOUNTING\_TOKEN
- MQC.MQSUB\_PROP\_PUBLICATION\_APPLICATIONID\_DATA

### **public MQAsyncStatus GetAsyncStatus();**

Emissions MQException

Renvoie un objet MQAsyncStatus qui représente l'activité asynchrone pour la connexion du gestionnaire de files d'attente.

### **public void Backout();**

Emet MQException.

Annuler les messages qui ont été lus ou écrits dans le point de synchronisation depuis le dernier point de synchronisation.

Les messages qui ont été écrits avec l'indicateur MQC.MQPMO\_SYNCPOINT défini sont supprimés des files d'attente. Les messages lus avec l'indicateur MQC.MQGMO\_SYNCPOINT sont réintégrés dans les files d'attente dont ils proviennent. Si les messages sont persistants, les modifications sont consignées.

Pour les clients reconnectables, le code anomalie MQRC\_NONE est renvoyé à un client une fois que la reconnexion a abouti.

### **public void Begin();**

Emet MQException.

Begin est pris en charge uniquement en mode liaisons de serveur. Il démarre une unité de travail globale.

### **public void Commit();**

Emet MQException.

Validez tous les messages qui ont été lus ou écrits dans le point de synchronisation depuis le dernier point de synchronisation.

Les messages écrits avec l'indicateur MQC.MQPMO\_SYNCPOINT défini sont mis à la disposition d'autres applications. Les messages extraits avec l'indicateur MQC.MQGMO\_SYNCPOINT sont supprimés. Si les messages sont persistants, les modifications sont consignées.

Les codes anomalie suivants sont renvoyés à un client reconnectable:

- MQRC\_CALL\_INTERRUPTED si la connexion est perdue lors de l'exécution de l'appel de validation.
- MQRC\_BACKED\_OUT si l'appel de validation est émis après la reconnexion.

### **Disconnect();**

Emet MQException.

Fermez la connexion au gestionnaire de files d'attente. Tous les objets accessibles sur ce gestionnaire de files d'attente ne sont plus accessibles à cette application. Pour accéder à nouveau aux objets, créez un objet MQQueueManager .

En règle générale, tout travail effectué dans le cadre d'une unité de travail est validé. Toutefois, si l'unité de travail est gérée par .NET, elle peut être annulée.

### **public void Put(int type, string destinationName, MQMessage message);**

### **public void Put(int type, string destinationName, MQMessage message**

### **MQPutMessageOptions putMessageOptions);**

### **public void Put(int type, string destinationName, string queueManagerName,**

### **string topicString, MQMessage message);**

### **public void Put(string queueName, MQMessage message);**

### **public void Put(string queueName, MQMessage message, MQPutMessageOptions**

### **putMessageOptions);**

### **public void Put(string queueName, string queueManagerName, MQMessage message);**

### **public void Put(string queueName, string queueManagerName, MQMessage message,**

### **MQPutMessageOptions putMessageOptions);**

### **public void Put(string queueName, string queueManagerName, MQMessage message,**

### **MQPutMessageOptions putMessageOptions, string alternateUserId);**

Emet MQException.

Place un message unique dans une file d'attente ou une rubrique sans créer d'objet MQQueue ou MQTopic au préalable.

#### **queueName**

Nom de la file d'attente dans laquelle placer le message.

**destinationName**

Nom d'un objet de destination. Il s'agit d'une file d'attente ou d'une rubrique en fonction de la valeur de *type*.

**type**

Type de l'objet de destination. Vous ne devez pas combiner les options.

**MQC.MQOT\_Q**

File d'attente

**MQC.MQOT\_TOPIC**

Topic

**QueueManagerName**

Nom du gestionnaire de files d'attente ou de l'alias du gestionnaire de files d'attente sur lequel la file d'attente est définie. Si le type MQC.MQOT\_TOPIC est spécifié, ce paramètre est ignoré.

Si la file d'attente est une file d'attente modèle et que le nom du gestionnaire de files d'attente résolu n'est pas ce gestionnaire de files d'attente, une exception MQException est émise.

**topicString**

*topicString* est combiné avec le nom de la rubrique dans l'objet de rubrique *destinationName*.

*topicString* est ignoré si *destinationName* est une file d'attente.

**Message**

Message à envoyer. Le message est un objet d'entrée/sortie.

Les codes anomalie suivants sont renvoyés à un client reconnectable:

- MQRC\_CALL\_INTERRUPTED si la connexion est interrompue lors de l'exécution d'un appel Put sur un message persistant.
- MQRC\_NONE si la connexion aboutit lors de l'exécution d'un appel Put sur un message non persistant (voir [Application Recovery](#)).

**Options de putMessage**

Options contrôlant les actions de l'insertion.

Si vous omettez *putMessageOptions*, une instance par défaut de *putMessageOptions* est créée. *putMessageOptions* est un objet d'entrée-sortie.

Si vous utilisez l'option MQPMO\_LOGICAL\_ORDER dans un client reconnectable, le code anomalie MQRC\_RECONNECT\_INCOMPATIBLE est renvoyé.

**AlternateUserid**

Indique un autre ID utilisateur utilisé pour vérifier l'autorisation lors du placement du message dans une file d'attente.

Vous pouvez omettre *alternateUserId* si vous ne définissez pas MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY dans *putMessageOptions*. Si vous définissez MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY, vous devez également définir *alternateUserId*. *alternateUserId* n'a pas d'effet sauf si vous avez également défini MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY.

## Constructeurs

```
public MQQueueManager();  
public MQQueueManager(string queueManagerName);  
public MQQueueManager(string queueManagerName, Int options);  
public MQQueueManager(string queueManagerName, Int options, string channel,  
string connName);  
public MQQueueManager(string queueManagerName, string channel, string  
connName);  
public MQQueueManager(string queueManagerName, System.Collections.Hashtable  
properties);
```

Emet MQException.

Crée une connexion à un gestionnaire de files d'attente. Sélectionnez entre la création d'une connexion client ou d'une connexion serveur.

Vous devez disposer du droit d'interrogation (inq) sur le gestionnaire de files d'attente lorsque vous tentez de vous connecter au gestionnaire de files d'attente. Sans droit d'interrogation, la tentative de connexion échoue.

Une connexion client est créée si l'une des conditions suivantes est vérifiée:

1. *channel* ou *connName* sont spécifiés dans le constructeur.
2. *HostName*, *Port* ou *Channel* sont spécifiés dans *properties*.
3. *MQEnvironment.HostName*, *MQEnvironment.Port* ou *MQEnvironment.Channel* sont spécifiés.

Les valeurs des propriétés de connexion sont définies par défaut dans l'ordre indiqué. Les valeurs *channel* et *connName* du constructeur sont prioritaires sur les valeurs de propriété du constructeur. Les valeurs de propriété du constructeur sont prioritaires sur les propriétés *MQEnvironment*.

Le nom d'hôte, le nom de canal et le port sont définis dans la classe *MQEnvironment*.

### QueueManagerName

Nom du gestionnaire de files d'attente ou du groupe de gestionnaires de files d'attente auquel se connecter.

Omettez le paramètre, ou laissez-le null ou vide pour effectuer une sélection de gestionnaire de files d'attente par défaut. La connexion de gestionnaire de files d'attente par défaut sur un serveur correspond au gestionnaire de files d'attente par défaut sur le serveur. La connexion de gestionnaire de files d'attente par défaut sur une connexion client est établie avec le gestionnaire de files d'attente auquel le programme d'écoute est connecté.

### options

Spécifiez les options de connexion MQCNO. Les valeurs doivent être applicables au type de connexion en cours d'établissement. Par exemple, si vous spécifiez les propriétés de connexion serveur suivantes pour une connexion client, une exception MQException est émise.

- MQC.MQCNO\_FASTPATH\_BINDING
- MQC.MQCNO\_STANDARD\_BINDING

### propriétés

Le paramètre *properties* prend une série de paires clé / valeur qui remplacent les propriétés définies par *MQEnvironment*; voir l'exemple «Remplacer les propriétés *MQEnvironment*», à la page 1858. Les propriétés suivantes peuvent être remplacées:

- MQC.CONNECT\_OPTIONS\_PROPERTY
- MQC.CONNECTION\_NAME\_PROPERTY
- MQC.ENCRYPTION\_POLICY\_SUITE\_B
- MQC.HOST\_NAME\_PROPERTY
- MQC.PORT\_PROPERTY

- MQC.CHANNEL\_PROPERTY
- MQC.SSL\_CIPHER\_SPEC\_PROPERTY
- MQC.SSL\_PEER\_NAME\_PROPERTY
- MQC.SSL\_CERT\_STORE\_PROPERTY
- MQC.SSL\_CRYPTO\_HARDWARE\_PROPERTY
- MQC.SECURITY\_EXIT\_PROPERTY
- MQC.SECURITY\_USERDATA\_PROPERTY
- MQC.SEND\_EXIT\_PROPERTY
- MQC.SEND\_USERDATA\_PROPERTY
- MQC.RECEIVE\_EXIT\_PROPERTY
- MQC.RECEIVE\_USERDATA\_PROPERTY
- MQC.USER\_ID\_PROPERTY
- MQC.PASSWORD\_PROPERTY
- MQC.MQAIR\_ARRAY
- MQC.KEY\_RESET\_COUNT
- MQC.FIPS\_REQUIRED
- MQC.HDR\_CMP\_LIST
- MQC.MSG\_CMP\_LIST
- MQC.TRANSPORT\_PROPERTY

#### **canal**

Nom d'un canal de connexion serveur

#### **connName**

Nom de connexion au format *HostName (Port)*.

Vous pouvez fournir une liste de *noms d'hôte* et de *ports* en tant qu'argument au constructeur MQQueueManager (String queueManagerName, Hashtable properties) à l'aide de CONNECTION\_NAME\_PROPERTY.

Exemple :

```

ConnectionName = "fred.mq.com(2344),nick.mq.com(3746),tom.mq.com(4288)";
Hashtable Properties=new Hashtable();
properties.Add(MQC.CONNECTION_NAME_PROPERTY,ConnectionName);
MQQueueManager qmgr=new MQQueue Manager("qmgrname",properties);

```

Lorsqu'une tentative de connexion est effectuée, la liste des noms de connexion est traitée dans l'ordre. Si la tentative de connexion au premier nom d'hôte et au premier port échoue, la connexion à la deuxième paire d'attributs est tentée. Le client répète ce processus jusqu'à ce qu'une connexion soit établie ou que la liste soit épuisée. Si la liste est épuisée, un code anomalie et un code achèvement appropriés sont renvoyés à l'application client.

Lorsqu'aucun numéro de port n'est fourni pour le nom de connexion, le port par défaut (configuré dans mqclient.ini) est utilisé.

### **Définition de la liste de connexions**

Vous pouvez définir la liste de connexion à l'aide des méthodes suivantes lorsque les options de reconnexion automatique du client sont définies:

#### **Définition de la liste de connexions via MQSERVER**

Vous pouvez définir la liste de connexion à l'aide de l'invite de commande.



A l'invite de commande, définissez la commande suivante:

```
MQSERVER=SYSTEM.DEF.SVRCONN/TCP/Hostname1(Port1),Hostname2(Por2),Hostname3(Port3)
```

Exemple :

```
MQSERVER=SYSTEM.DEF.SVRCONN/TCP/fred.mq.com(5266),nick.mq.com(6566),jack.mq.com(8413)
```

Si vous définissez la connexion dans MQSERVER, ne la définissez pas dans l'application.

Si vous définissez la liste de connexions dans l'application, l'application écrase tout ce qui est défini dans la variable d'environnement MQSERVER.

### Définir la liste de connexion via l'application

Vous pouvez définir la liste de connexion dans l'application en spécifiant le nom d'hôte et les propriétés de port.

```
String connName = "fred.mq.com(2344), nick.mq.com(3746), chris.mq.com(4288)";  
MQQueueManager qm = new MQQueueManager("QM1", "TestChannel", connName);
```

### Définition de la liste de connexion via app.config

App.config est un fichier XML dans lequel vous spécifiez les paires clé-valeur.

Dans la liste de connexion, indiquez

```
<app.Settings>  
<add key="Connection1" value="Hostname1(Port1)"/>  
<add key="Connection2" value="Hostname2(Port2)"/>  
</app.Settings>
```

Exemple :

```
<app.Settings>  
<add key="Connection1" value="fred.mq.com(2966)"/>  
<add key="Connection2" value="alex.mq.com(6533)"/>  
</app.Settings>
```

Vous pouvez modifier directement la liste de connexion dans le fichier app.config.

### Définition de la liste de connexion via MQEnvironment

Pour définir la liste de connexions via MQEnvironment, utilisez la propriété *ConnectionName*.

```
MQEnvironment.ConnectionName = "fred.mq.com(4288),alex.mq.com(5211);
```

La propriété *ConnectionName* remplace les propriétés de nom d'hôte et de port définies dans MQEnvironment.

### Créer une connexion client

L'exemple suivant montre comment créer une connexion client à un gestionnaire de files d'attente. Vous pouvez créer une connexion client en définissant les variables MQEnvironment avant de créer un objet MQQueueManager.

```

MQEnvironment.Hostname = "fred.mq.com"; // host to connect to
MQEnvironment.Port     = 1414;         // port to connect to
                                     // If not explicitly set,
                                     // defaults to 1414
                                     // (the default IBM MQ port)
MQEnvironment.Channel  = "channel.name"; // the case sensitive
                                     // name of the
                                     // SVR CONN channel on
                                     // the queue manager
MQQueueManager qMgr    = new MQQueueManager("MYQM");

```

Figure 11. Connexion client

## Remplacer les propriétés MQEnvironment

L'exemple suivant montre comment créer un gestionnaire de files d'attente avec son ID utilisateur et son mot de passe définis dans une table de hachage.

```

Hashtable properties = new Hashtable();

properties.Add( MQC.USER_ID_PROPERTY, "ExampleUserId" );
properties.Add( MQC.PASSWORD_PROPERTY, "ExamplePassword" );

try
{
    MQQueueManager qMgr = new MQQueueManager("qmgrname", properties);
}
catch (MQException mqe)
{
    System.Console.WriteLine("Connect failed with " + mqe.Message);
    return((int)mqe.Reason);
}

```

Figure 12. Remplacement des propriétés MQEnvironment

## Créer une connexion reconnectable

L'exemple suivant montre comment reconnecter automatiquement un client à un gestionnaire de files d'attente.

```

Hashtable properties = new Hashtable(); // The queue manager name and the
                                     // properties how it has to be connected

properties.Add(MQC.CONNECT_OPTIONS_PROPERTY, MQC.MQCNO_RECONNECT); // Options
                                     // through which reconnection happens

properties.Add(MQC.CONNECTION_NAME_PROPERTY, "fred.mq.com(4789),nick.mq.com(4790)"); // The list
                                     // of queue managers through which reconnection happens

MQ QueueManager qmgr = new MQQueueManager("qmgrname", properties);

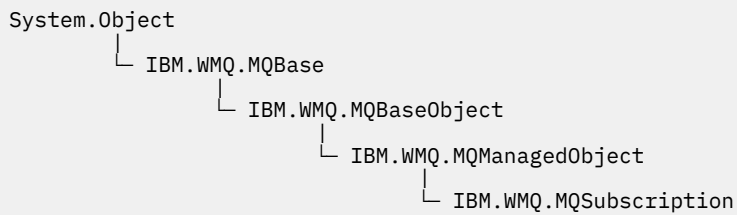
```

Figure 13. Reconnexion automatique d'un client à un gestionnaire de files d'attente

## Classe MQSubscription.NET

Utilisez MQSubscription pour demander que les publications conservées soient envoyées à l'abonné. MQSubscription est une propriété d'un objet MQTopic ouvert pour abonnement.

### Classe



```
public class IBM.WMQ.MQSubscription extends IBM.WMQ.MQManagedObject;
```

- «Propriétés», à la page 1859
- «Méthodes», à la page 1859
- «Constructeurs», à la page 1859

## Propriétés

Accédez aux propriétés d'abonnement à l'aide de la classe `MQManagedObject` ; voir «Propriétés», à la page 1816.

## Méthodes

Accédez aux méthodes d'abonnement `Inquire`, `Set` et `Get` à l'aide de la classe `MQManagedObject` ; voir «Méthodes», à la page 1817.

### **public int RequestPublicationUpdate(int options);**

Emet `MQException`.

Demander une publication mise à jour pour la rubrique en cours. Si le gestionnaire de files d'attente possède des publications conservées pour la rubrique, elles sont envoyées à l'abonné.

Avant d'appeler `RequestPublicationUpdate`, ouvrez une rubrique pour l'abonnement afin d'obtenir un objet `MQSubscription` .

En règle générale, ouvrez l'abonnement avec l'option `MQC.MQSO_PUBLICATIONS_ON_REQUEST` . Si aucun caractère générique n'est présent dans la chaîne de rubrique, une seule publication est envoyée suite à cet appel. Si la chaîne de rubrique contient des caractères génériques, de nombreuses publications peuvent être envoyées. La méthode renvoie le nombre de publications conservées qui sont envoyées à la file d'attente d'abonnement. Il n'est pas garanti que ces nombreuses publications soient reçues, surtout s'il s'agit de messages non persistants.

### options

#### **MQC.MQSRO\_FAIL\_IF QUIESCING**

La méthode échoue si le gestionnaire de files d'attente est à l'état de repos. Sous z/OS, pour une application CICS ou IMS , `MQC.MQSRO_FAIL_IF QUIESCING` force également la méthode à échouer si la connexion est à l'état de repos.

#### **MQC.MQSRO\_NONE**

Aucune option n'est spécifiée.

## Constructeurs

Aucun constructeur public .

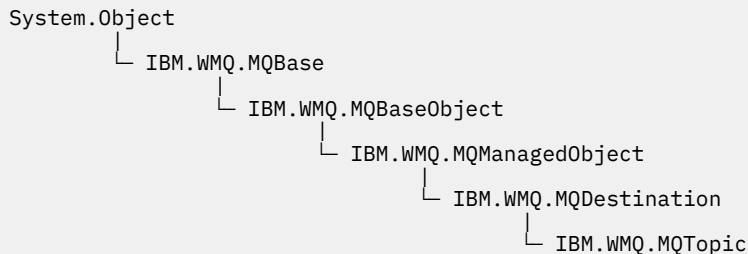
Un objet `MQSubscription` est renvoyé dans la propriété `SubscriptionReference` d'un objet `MQTopic` ouvert pour l'abonnement,

Appelez la méthode `RequestPublicationUpdate` . `MQSubscription` est une sous-classe de `MQManagedObject`. Utilisez la référence pour accéder aux propriétés et aux méthodes de `MQManagedObject`.

## Classe MQTopic.NET

Utilisez MQTopic pour publier ou abonner des messages sur une rubrique, ou pour interroger ou définir des attributs d'une rubrique. Créez un objet MQTopic pour la publication ou l'abonnement à l'aide d'un constructeur ou de la méthode MQQueueManager.AccessTopic .

### Classe



```
public class IBM.WMQ.MQTopic extends IBM.WMQ.MQDestination;
```

- «Propriétés», à la page 1860
- «Méthodes», à la page 1860
- «Constructeurs», à la page 1862

### Propriétés

Test de l'émission de MQException lors de l'obtention des propriétés.

#### **public Boolean IsDurable {get;}**

Propriété en lecture seule qui renvoie True si l'abonnement est durable ou False dans le cas contraire. Si la rubrique a été ouverte pour publication, la propriété est ignorée et renvoie toujours False.

#### **public Boolean IsManaged {get;};**

Propriété en lecture seule qui renvoie True si l'abonnement est géré par le gestionnaire de files d'attente ou False dans le cas contraire. Si la rubrique a été ouverte pour publication, la propriété est ignorée et renvoie toujours False.

#### **public Boolean IsSubscribed {get;};**

Propriété en lecture seule qui renvoie True si la rubrique a été ouverte pour abonnement et False si la rubrique a été ouverte pour publication.

#### **public MQSubscription SubscriptionReference {get;};**

Propriété en lecture seule qui renvoie l'objet MQSubscription associé à un objet de rubrique ouvert pour abonnement. La référence est disponible si vous souhaitez modifier les options de fermeture ou démarrer l'une des méthodes d'objet.

#### **public MQDestination UnmanagedDestinationReference {get;};**

Propriété en lecture seule qui renvoie le MQQueue associé à un abonnement non géré. Il s'agit de la destination spécifiée lors de la création de l'objet de rubrique. La propriété renvoie la valeur null pour tous les objets de rubrique ouverts pour publication ou avec un abonnement géré.

### Méthodes

#### **public void Put(MQMessage message);**

#### **public void Put(MQMessage message, MQPutMessageOptions putMessageOptions);**

Emet une exception MQException.

Publie un message dans le sujet.

Les modifications apportées à l'objet `MQMessage` après l'appel d'insertion n'affectent pas le message réel dans la file d'attente IBM MQ ou dans la rubrique de publication.

`Put` met à jour les propriétés `MessageId` et `CorrelationId` de l'objet `MQMessage` et n'efface pas les données de message. Les appels `Put` ou `Get` supplémentaires font référence aux informations mises à jour dans l'objet `MQMessage`. Par exemple, dans le fragment de code suivant, le premier message contient a et le second ab.

```
msg.WriteString("a");
q.Put(msg, pmo);
msg.WriteString("b");
q.Put(msg, pmo);
```

### Message

Objet `MQMessage` contenant les données du descripteur de message et le message à envoyer. Le descripteur de message peut être modifié à la suite de cette méthode. Les valeurs du descripteur de message immédiatement après la fin de cette méthode sont les valeurs qui ont été placées dans la file d'attente ou publiées dans la rubrique.

Les codes anomalie suivants sont renvoyés à un client reconnectable:

- `MQRC_CALL_INTERRUPTED` si la connexion est interrompue lors de l'exécution d'un appel `Put` sur un message persistant et que la reconnexion a abouti.
- `MQRC_NONE` si la connexion aboutit lors de l'exécution d'un appel `Put` sur un message non persistant (voir [Application Recovery](#)).

### Options de `putMessage`

Options contrôlant l'action de l'insertion.

Si `putMessageOptions` n'est pas spécifié, l'instance par défaut de `MQPutMessageOptions` est utilisée.

Si vous utilisez l'option `MQPMO_LOGICAL_ORDER` dans un client reconnectable, le code anomalie `MQRC_RECONNECT_INCOMPATIBLE` est renvoyé.

**Remarque :** Pour plus de simplicité et de performances, si vous souhaitez placer un seul message dans une file d'attente, utilisez l'objet `MQQueueManager.Put`. Vous devez disposer d'un objet `MQQueue` pour cela.

```
public void Get(MQMessage message);  
public void Get(MQMessage message, MQGetMessageOptions getMessageOptions);  
public void Get(MQMessage message, MQGetMessageOptions getMessageOptions, int  
MaxMsgSize);
```

Emet une exception `MQException`.

Extrait un message de la rubrique.

Cette méthode utilise une instance par défaut de `MQGetMessageOptions` pour effectuer l'extraction. L'option de message utilisée est `MQGMO_NOWAIT`.

Si l'extraction échoue, l'objet `MQMessage` n'est pas modifié. Si elle aboutit, le descripteur de message et les parties de données de message du `MQMessage` sont remplacés par le descripteur de message et les données de message du message entrant.

Tous les appels à IBM MQ à partir d'un `MQQueueManager` particulier sont synchrones. Par conséquent, si vous effectuez une opération `get` avec attente, toutes les autres unités d'exécution utilisant le même `MQQueueManager` ne peuvent pas effectuer d'autres appels IBM MQ jusqu'à ce que l'appel `Get` soit effectué. Si vous avez besoin de plusieurs unités d'exécution pour accéder à IBM MQ simultanément, chaque unité d'exécution doit créer son propre objet `MQQueueManager`.

### Message

Contient le descripteur de message et les données de message renvoyées. Certaines des zones du descripteur de message sont des paramètres d'entrée. Il est important de s'assurer que les paramètres d'entrée `MessageId` et `CorrelationId` sont définis comme requis.

Un client reconnectable renvoie le code anomalie MQRC\_BACKED\_OUT après une reconnexion réussie, pour les messages reçus sous MQGM\_SYNCPOINT.

### Options de getMessage

Options contrôlant l'action de la commande get.

L'utilisation de l'option MQC.MQGM\_CONVERT peut générer une exception avec le code anomalie MQC.MQRC\_CONVERTED\_STRING\_TOO\_BIG lors de la conversion de codes de caractères codés sur un octet en codes codés sur deux octets. Dans ce cas, le message est copié dans la mémoire tampon sans conversion.

Si *getMessageOptions* n'est pas spécifié, l'option de message utilisée est MQGM\_NOWAIT.

Si vous utilisez l'option MQGM\_LOGICAL\_ORDER dans un client reconnectable, le code anomalie MQRC\_RECONNECT\_INCOMPATIBLE est renvoyé.

### Taille MaxMsg

Message le plus volumineux que cet objet message doit recevoir. Si le message de la file d'attente est supérieur à cette taille, l'un des deux événements suivants se produit:

- Si l'indicateur MQGM\_ACCEPT\_TRUNCATED\_MSG est défini dans l'objet MQGetMessageOptions, le message est rempli avec autant de données de message que possible. Une exception est émise avec le code achèvement MQCC\_WARNING et le code anomalie MQRC\_TRUNCATED\_MSG\_ACCEPTED.
- Si l'indicateur MQGM\_ACCEPT\_TRUNCATED\_MSG n'est pas défini, le message reste dans la file d'attente. Une exception est émise avec le code achèvement MQCC\_WARNING et le code anomalie MQRC\_TRUNCATED\_MSG\_FAILED.

Si *MaxMsgSize* n'est pas spécifié, l'intégralité du message est extraite.

## Constructeurs

```
public MQTopic(MQQueueManager queueManager, MQDestination destination, string
topicName, string topicObject, int options);
public MQTopic(MQQueueManager queueManager, MQDestination destination, string
topicName, string topicObject, int options, string alternateUserId);
public MQTopic(MQQueueManager queueManager, MQDestination destination, string
topicName, string topicObject, int options, string alternateUserId, string
subscriptionName);
public MQTopic(MQQueueManager queueManager, MQDestination destination, string
topicName, string topicObject, int options, string alternateUserId, string
subscriptionName, System.Collections.Hashtable properties);
public MQTopic(MQQueueManager queueManager, string topicName, string
topicObject, int openAs, int options);
public MQTopic(MQQueueManager queueManager, string topicName, string
topicObject, int openAs, int options, string alternateUserId);
public MQTopic(MQQueueManager queueManager, string topicName, string
topicObject, int options, string alternateUserId, string subscriptionName);
public MQTopic(MQQueueManager queueManager, string topicName, string
topicObject, int options, string alternateUserId, string subscriptionName,
System.Collections.Hashtable properties);
```

Accédez à une rubrique sur *queueManager*.

Les objets MQTopic sont étroitement liés aux objets de rubrique d'administration, qui sont parfois appelés objets de rubrique. En entrée, topicObject pointe vers un objet de rubrique d'administration. Le constructeur MQTopic obtient une chaîne de rubrique de l'objet de rubrique et la combine avec topicName pour créer un nom de rubrique. topicObject ou topicName peut être null. Le nom de rubrique est mis en correspondance avec l'arborescence de rubriques et le nom de l'objet de rubrique d'administration correspondant le plus proche est renvoyé dans topicObject.

Les rubriques associées à l'objet `MQTopic` sont le résultat de la combinaison de deux chaînes de rubrique. La première chaîne de rubrique est définie par l'objet de rubrique d'administration identifié par `topicObject`. La deuxième chaîne de rubrique est `topicString`. La chaîne de rubrique résultante associée à l'objet `MQTopic` peut identifier plusieurs rubriques en incluant des caractères génériques.

Selon que la rubrique est ouverte pour la publication ou l'abonnement, vous pouvez utiliser les méthodes `MQTopic.Put` pour la publication sur des rubriques ou les méthodes `MQTopic.Get` pour la réception de publications sur des rubriques. Si vous souhaitez publier et vous abonner à la même rubrique, vous devez accéder à la rubrique deux fois, une fois pour la publication et une fois pour l'abonnement.

Si vous créez un objet `MQTopic` pour l'abonnement, sans fournir d'objet `MQDestination`, un abonnement géré est supposé. Si vous transmettez une file d'attente en tant qu'objet `MQDestination`, un abonnement non géré est supposé. Vous devez vous assurer que les options d'abonnement que vous définissez sont cohérentes avec l'abonnement géré ou non géré.

### **queueManager**

Gestionnaire de files d'attente sur lequel accéder à une rubrique.

### **destination**

`destination` est une instance `MQQueue`. En fournissant `destination`, `MQTopic` est ouvert en tant qu'abonnement non géré. Les publications de la rubrique sont distribuées dans la file d'attente accessible en tant que `destination`.

### **topicName**

Chaîne de rubrique correspondant à la deuxième partie du nom de la rubrique. `topicName` est concaténé avec la chaîne de rubrique définie dans l'objet de rubrique d'administration `topicObject`. Vous pouvez définir `topicName` sur null, auquel cas le nom de rubrique est défini par la chaîne de rubrique dans `topicObject`.

### **topicObject**

En entrée, `topicObject` est le nom de l'objet de rubrique qui contient la chaîne de rubrique qui forme la première partie du nom de rubrique. La chaîne de rubrique dans `topicObject` est concaténée avec `topicName`. Les règles de construction des chaînes de rubrique sont définies dans [Combinaison de chaînes de rubrique](#).

Dans la sortie, `topicObject` contient le nom de l'objet de rubrique d'administration qui correspond le plus à la rubrique identifiée par la chaîne de rubrique dans l'arborescence de rubriques.

### **openAs**

Accédez à la rubrique pour la publication ou l'abonnement. Le paramètre ne peut contenir qu'une seule des options suivantes:

- `MQC.MQTOPIC_OPEN_AS_SUBSCRIPTION`
- `MQC.MQTOPIC_OPEN_AS_PUBLICATION`

### **options**

Combinez les options qui contrôlent l'ouverture de la rubrique pour la publication ou l'abonnement. Utilisez les constantes `MQC.MQSO_*` pour accéder à une rubrique pour l'abonnement et les constantes `MQC.MQOO_*` pour accéder à une rubrique pour la publication.

Si plusieurs options sont requises, ajoutez les valeurs ensemble ou combinez les valeurs d'option à l'aide de l'opérateur OR bit à bit.

### **AlternateUserid**

Indiquez l'ID utilisateur de remplacement utilisé pour vérifier l'autorisation requise pour terminer l'opération. Vous devez spécifier `alternateUserId`, si `MQC.MQOO_ALTERNATE_USER_AUTHORITY` ou `MQC.MQSO_ALTERNATE_USER_AUTHORITY` est défini dans le paramètre d'options.

## subscriptionName

*subscriptionName* est obligatoire si les options MQC.MQSO\_DURABLE ou MQC.MQSO\_ALTER sont fournies. Dans les deux cas, MQTopic est implicitement ouvert pour l'abonnement. Une exception est émise si MQC.MQSO\_DURABLE est défini et que l'abonnement existe, ou si MQC.MQSO\_ALTER est défini, et que l'abonnement n'existe pas.

## propriétés

Définissez les propriétés d'abonnement spéciales répertoriées à l'aide d'une table de hachage. Les entrées spécifiées dans la table de hachage sont mises à jour avec les valeurs de sortie. Les entrées ne sont pas ajoutées à la table de hachage pour signaler les valeurs de sortie.

- MQC.MQSUB\_PROP\_ALTERNATE\_SECURITY\_ID
- MQC.MQSUB\_PROP\_SUBSCRIPTION\_EXPIRY
- MQC.MQSUB\_PROP\_SUBSCRIPTION\_USER\_DATA
- MQC.MQSUB\_PROP\_SUBSCRIPTION\_CORRELATION\_ID
- MQC.MQSUB\_PROP\_PUBLICATION\_PRIORITY
- MQC.MQSUB\_PROP\_PUBLICATION\_ACCOUNTING\_TOKEN
- MQC.MQSUB\_PROP\_PUBLICATION\_APPLICATIONID\_DATA

```
public MQTopic MQQueueManager.AccessTopic(MQDestination destination, string
topicName, string topicObject, int options);
public MQTopic MQQueueManager.AccessTopic(MQDestination destination, string
topicName, string topicObject, int options, string alternateUserId);
public MQTopic MQQueueManager.AccessTopic(MQDestination destination, string
topicName, string topicObject, int options, string alternateUserId, string
subscriptionName);
public MQTopic MQQueueManager.AccessTopic(MQDestination destination, string
topicName, string topicObject, int options, string alternateUserId, string
subscriptionName, System.Collections.Hashtable properties);
public MQTopic MQQueueManager.AccessTopic(string topicName, string topicObject,
int openAs, int options);
public MQTopic MQQueueManager.AccessTopic(string topicName, string topicObject,
int openAs, int options, string alternateUserId);
public MQTopic MQQueueManager.AccessTopic(string topicName, string topicObject,
int options, string alternateUserId, string subscriptionName);
public MQTopic MQQueueManager.AccessTopic(string topicName, string
topicObject, int options, string alternateUserId, string subscriptionName,
System.Collections.Hashtable properties);
```

Accédez à une rubrique de ce gestionnaire de files d'attente.

Les objets MQTopic sont étroitement liés aux objets de rubrique d'administration, qui sont parfois appelés objets de rubrique. En entrée, topicObject pointe vers un objet de rubrique d'administration. Le constructeur MQTopic obtient une chaîne de rubrique de l'objet de rubrique et la combine avec topicName pour créer un nom de rubrique. topicObject ou topicName peut être null. Le nom de rubrique est mis en correspondance avec l'arborescence de rubriques et le nom de l'objet de rubrique d'administration correspondant le plus proche est renvoyé dans topicObject.

Les rubriques associées à l'objet MQTopic sont le résultat de la combinaison de deux chaînes de rubrique. La première chaîne de rubrique est définie par l'objet de rubrique d'administration identifié par topicObject. La deuxième chaîne de rubrique est topicString. La chaîne de rubrique résultante associée à l'objet MQTopic peut identifier plusieurs rubriques en incluant des caractères génériques.

Selon que la rubrique est ouverte pour la publication ou l'abonnement, vous pouvez utiliser les méthodes MQTopic.Put pour la publication sur des rubriques ou les méthodes MQTopic.Get pour la réception de publications sur des rubriques. Si vous souhaitez publier et vous abonner à la même rubrique, vous devez accéder à la rubrique deux fois, une fois pour la publication et une fois pour l'abonnement.



Si vous créez un objet `MQTopic` pour l'abonnement, sans fournir d'objet `MQDestination`, un abonnement géré est supposé. Si vous transmettez une file d'attente en tant qu'objet `MQDestination`, un abonnement non géré est supposé. Vous devez vous assurer que les options d'abonnement que vous définissez sont cohérentes avec l'abonnement géré ou non géré.

### **destination**

*destination* est une instance `MQQueue`. En fournissant *destination*, `MQTopic` est ouvert en tant qu'abonnement non géré. Les publications de la rubrique sont distribuées dans la file d'attente accessible en tant que *destination*.

### **topicName**

Chaîne de rubrique correspondant à la deuxième partie du nom de la rubrique. *topicName* est concaténé avec la chaîne de rubrique définie dans l'objet de rubrique d'administration *topicObject*. Vous pouvez définir *topicName* sur null, auquel cas le nom de rubrique est défini par la chaîne de rubrique dans *topicObject*.

### **topicObject**

En entrée, *topicObject* est le nom de l'objet de rubrique qui contient la chaîne de rubrique qui forme la première partie du nom de rubrique. La chaîne de rubrique dans *topicObject* est concaténée avec *topicName*. Les règles de construction des chaînes de rubrique sont définies dans [Combinaison de chaînes de rubrique](#).

Dans la sortie, *topicObject* contient le nom de l'objet de rubrique d'administration qui correspond le plus à la rubrique identifiée par la chaîne de rubrique dans l'arborescence de rubriques.

### **openAs**

Accédez à la rubrique pour la publication ou l'abonnement. Le paramètre ne peut contenir qu'une seule des options suivantes:

- `MQC.MQTOPIC_OPEN_AS_SUBSCRIPTION`
- `MQC.MQTOPIC_OPEN_AS_PUBLICATION`

### **options**

Combinez les options qui contrôlent l'ouverture de la rubrique pour la publication ou l'abonnement. Utilisez les constantes `MQC.MQSO_*` pour accéder à une rubrique pour l'abonnement et les constantes `MQC.MQOO_*` pour accéder à une rubrique pour la publication.

Si plusieurs options sont requises, ajoutez les valeurs ensemble ou combinez les valeurs d'option à l'aide de l'opérateur OR bit à bit.

### **AlternateUserid**

Indiquez l'ID utilisateur de remplacement utilisé pour vérifier l'autorisation requise pour terminer l'opération. Vous devez spécifier *alternateUserId*, si `MQC.MQOO_ALTERNATE_USER_AUTHORITY` ou `MQC.MQSO_ALTERNATE_USER_AUTHORITY` est défini dans le paramètre d'options.

### **subscriptionName**

*subscriptionName* est obligatoire si les options `MQC.MQSO_DURABLE` ou `MQC.MQSO_ALTER` sont fournies. Dans les deux cas, `MQTopic` est implicitement ouvert pour l'abonnement. Une exception est émise si `MQC.MQSO_DURABLE` est défini et que l'abonnement existe, ou si `MQC.MQSO_ALTER` est défini, et que l'abonnement n'existe pas.

### **propriétés**

Définissez les propriétés d'abonnement spéciales répertoriées à l'aide d'une table de hachage. Les entrées spécifiées dans la table de hachage sont mises à jour avec les valeurs de sortie. Les entrées ne sont pas ajoutées à la table de hachage pour signaler les valeurs de sortie.

- `MQC.MQSUB_PROP_ALTERNATE_SECURITY_ID`
- `MQC.MQSUB_PROP_SUBSCRIPTION_EXPIRY`
- `MQC.MQSUB_PROP_SUBSCRIPTION_USER_DATA`
- `MQC.MQSUB_PROP_SUBSCRIPTION_CORRELATION_ID`

- MQC.MQSUB\_PROP\_PUBLICATION\_PRIORITY
- MQC.MQSUB\_PROP\_PUBLICATION\_ACCOUNTING\_TOKEN
- MQC.MQSUB\_PROP\_PUBLICATION\_APPLICATIONID\_DATA

## Interface IMQObjectTrigger.NET

Implémentez IMQObjectTrigger pour traiter les messages transmis par le moniteur **runmqdmn**.NET .

### Interface

```
public interface IBM.WMQMonitor.IMQObjectTrigger();
```

Selon que le contrôle de point de synchronisation est spécifié dans la commande **runmqdmn** , le message est supprimé de la file d'attente avant ou après le retour de la méthode Execute .

### Méthodes

**void Execute (MQQueueManager queueManager, MQQueue queue, MQMessage message, string param);**

**queueManager**

Gestionnaire de files d'attente hébergeant la file d'attente surveillée.

**file d'attente**

File d'attente surveillée.

**Message**

Message lu à partir de la file d'attente.

**paramètre**

Données transmises à partir de UserParameter.

## Interface MQC.NET

Faites référence à une constante MQI en ajoutant le préfixe MQC . au nom de la constante. MQC définit toutes les constantes utilisées par MQI.

### Interface

```
System.Object
└─ IBM.WMQ.MQC
```

```
public interface IBM.WMQ.MQC extends System.Object;
```

### Exemple

```
MQQueue queue;
queue.closeOptions = MQC.MQCO_DELETE;
```

## Identificateurs de jeu de caractères pour les applications .NET

Description des jeux de caractères que vous pouvez sélectionner pour coder les messages .NET IBM MQ

Jeu de caractères	Description
37	ibm037

<b>Jeu de caractères</b>	<b>Description</b>
437	ibm437 / PC d'origine
500	ibm500
819	iso-8859-1 / latin1 / ibm819
1 200	Unicode
1208	UTF-8
273	ibm273
277	ibm277
278	ibm278
280	ibm280
284	ibm284
285	ibm285
297	ibm297
420	ibm420
424	ibm424
737	ibm737 / PC Grec
775	ibm775 / PC Baltique
813	iso-8859-7 / Greek / ibm813
838	ibm838
850	ibm850 / PC Latin 1
852	ibm852 / PC Latin 2
855	ibm855 / PC Cyrillique
856	ibm856
857	ibm857 / PC Turc
860	ibm860 / PC Portugais
861	ibm861 / PC islandais
862	ibm862 / PC Hébreu
863	ibm863 / PC Français canadien
864	ibm864 / PC Arabe
865	ibm865 / PC Nordic
866	ibm866 / PC Russe
868	ibm868
869	ibm869 / PC Grec moderne
870	ibm870
871	ibm871
874	ibm874

<b>Jeu de caractères</b>	<b>Description</b>
875	ibm875
912	iso-8859-2 / latin2 / ibm912
913	iso-8859-3 / latin3 / ibm913
914	iso-8859-4 / latin4 / ibm914
915	iso-8859-5 / cyrillique / ibm915
916	iso-8859-8 / hebrew / ibm916
918	ibm918
920	iso-8859-9 / latin5 / ibm920
921	ibm921
922	ibm922
930	ibm930
932	PC Japonais
933	ibm933
935	ibm935
937	ibm937
939	ibm939
942	ibm942
943	ibm943
948	ibm948
949	ibm949
950	ibm950 / Big 5 Chinois traditionnel
954	EUCJIS
964	ibm964 / CNS 11643 Chinois traditionnel
970	ibm970
1006	ibm1006
1025	ibm1025
1026	ibm1026
1089	iso-8859-6 / arabic / ibm1089
1097	ibm1097
1098	ibm1098
1112	ibm1112
1122	ibm1122
1123	ibm1123
1124	ibm1124
1250	Windows Latin 2

Jeu de caractères	Description
1251	Windows Cyrillique
1252	Windows Latin 1
1253	Windows Grec
1254	Windows Turc
1255	Windows Hébreu
1256	Windows Arabe
1257	Windows Baltique
1258	Windows Vietnamien
1381	ibm1381
1383	ibm1383
2022	norme JIS
5601	ksc-5601 Coréen
33722	ibm33722

## IBM MQ classes C++

Les classes C++ IBM MQ encapsulent l'interface MQI ( IBM MQ Message Queue Interface). Il existe un fichier d'en-tête C++ unique, **imqi.hpp**, qui couvre toutes ces classes.

Pour chaque classe, les informations suivantes sont affichées:

### Diagramme de hiérarchie de classes

Diagramme de classes montrant la classe dans sa relation d'héritage avec ses classes parent immédiates, le cas échéant.

### Autres classes pertinentes

Le document contient des liens vers d'autres classes pertinentes, telles que les classes parent, et les classes d'objets utilisées dans les signatures de méthode.

### Attributs d'objet

Attributs de la classe. Ces attributs s'ajoutent à ceux définis pour les classes parent. De nombreux attributs reflètent les membres de la structure de données IBM MQ (voir «[Références croisées C++ et MQI](#)», à la page 1871 ). Pour des descriptions détaillées, voir «[Attributs des objets](#)», à la page 835.

### Constructeurs

Signatures des méthodes spéciales utilisées pour créer un objet de la classe.

### Méthodes d'objet (public)

Signatures des méthodes qui nécessitent une instance de la classe pour leur fonctionnement et qui n'ont pas de restrictions d'utilisation.

Lorsqu'elle s'applique, les informations suivantes sont également affichées:

### Méthodes de classe (public)

Signatures des méthodes qui ne nécessitent pas d'instance de la classe pour leur fonctionnement et qui n'ont pas de restrictions d'utilisation.

### Méthodes surchargées (classe parent)

Signatures des méthodes virtuelles définies dans les classes parent, mais présentant un comportement polymorphe différent pour cette classe.

### **Méthodes d'objet (protégées)**

Signatures des méthodes qui nécessitent une instance de la classe pour leur opération et qui sont réservées pour être utilisées par les implémentations de classes dérivées. Cette section n'intéresse que les auteurs de classe, par opposition aux utilisateurs de classe.

### **Données d'objet (protégées)**

Détails d'implémentation des données d'instance d'objet disponibles pour les implémentations de classes dérivées. Cette section n'intéresse que les auteurs de classe, par opposition aux utilisateurs de classe.

### **Codes raison**

Les valeurs MQRC\_\* (voir [Codes d'achèvement et d'anomalie de l'API](#)) qui peuvent être attendues à partir des méthodes qui échouent. Pour obtenir une liste exhaustive des codes anomalie pouvant se produire pour un objet d'une classe, consultez la documentation de la classe parent. La liste documentée des codes raison d'une classe n'inclut pas les codes raison des classes parent.

### **Remarque :**

1. Les objets de ces classes ne sont pas sécurisés pour les unités d'exécution. Cela garantit des performances optimales, mais il est important de ne pas accéder à un objet à partir de plusieurs unités d'exécution.
2. Pour un programme à unités d'exécution multiples, il est recommandé d'utiliser un objet ImqQueueManager distinct pour chaque unité d'exécution. Chaque objet gestionnaire doit disposer de sa propre collection indépendante d'autres objets, ce qui garantit que les objets des différentes unités d'exécution sont isolés les uns des autres.

Ces classes sont les suivantes :

- [«Classe C++ d'enregistrement ImqAuthentication»](#), à la page 1888
- [«Classe C++ ImqBinary»](#), à la page 1890
- [«Classe C++ ImqCache»](#), à la page 1892
- [«Classe C++ ImqChannel»](#), à la page 1895
- [«Classe C++ imqCICSBridgeHeader»](#), à la page 1901
- [«Classe C++ ImqDeadLetterHeader»](#), à la page 1907
- [«ImqDistributionRépertoire la classe C++»](#), à la page 1910
- [«Classe C++ ImqError»](#), à la page 1911
- [«Classe C++ ImqGetMessageOptions»](#), à la page 1912
- [«Classe C++ ImqHeader»](#), à la page 1916
- [«Classe C++ imqIMSBridgeHeader»](#), à la page 1917
- [«Classe C++ ImqItem»](#), à la page 1920
- [«Classe C++ ImqMessage»](#), à la page 1922
- [«Classe C++ de la fonction de suivi ImqMessage»](#), à la page 1929
- [«Classe C++ ImqNamelist»](#), à la page 1932
- [«Classe C++ ImqObject»](#), à la page 1933
- [«Classe C++ ImqProcess»](#), à la page 1939
- [«Classe C++ ImqPutMessageOptions»](#), à la page 1941
- [«Classe C++ ImqQueue»](#), à la page 1943
- [«Classe C++ du gestionnaire ImqQueue»](#), à la page 1954
- [«Classe C++ d'en-tête ImqReference»](#), à la page 1971
- [«Classe C++ ImqString»](#), à la page 1973
- [«Classe C++ ImqTrigger»](#), à la page 1979
- [«Classe C++ d'en-tête ImqWork»](#), à la page 1981

## Références croisées C++ et MQI

Cette collection de rubriques contient des informations relatives à C++ pour l'interface MQI.

Lisez ces informations avec «Types de données utilisés dans l'interface MQI», à la page 237.

Cette table associe les structures de données MQI aux classes C++ et aux fichiers d'inclusion. Les rubriques suivantes présentent des informations de référence croisée pour chaque classe C++. Ces références croisées concernent l'utilisation des interfaces de procédure IBM MQ sous-jacentes. Les classes ImqBinary, ImqDistributionList et ImqString ne comportent aucun attribut appartenant à cette catégorie et sont exclues.

*Tableau 845. Référence croisée de structure de données, de classe et de fichier d'inclusion*

Structure de données	Classe	fichier include
MQAIR	Enregistrement ImqAuthentication	imqair.hpp
	ImqBinary	imqbin.hpp
	ImqCache	imqcac.hpp
MQCD	ImqChannel	imqchl.hpp
MQCIH	ImqCICSBridgeHeader	imqcih.hpp
MQDLH	ImqDeadLetterHeader	imqdlh.hpp
MQOR	Liste ImqDistribution	imqdst.hpp
	ImqError	imqerr.hpp
MQGMO	ImqGetMessageOptions	imqgmo.hpp
	ImqHeader	imqhdr.hpp
MQIIH	ImqIMSBridgeHeader	imqiih.hpp
	ImqItem	imqitm.hpp
MQMD	ImqMessage	imqmsg.hpp
	Dispositif de suivi ImqMessage	imqmtr.hpp
	ImqNamelist	imqnml.hpp
MQOD, MQRR	ImqObject	imqobj.hpp
MQPMO, MQPMR, MQRR	ImqPutMessageOptions	imqpmo.hpp
	ImqProcess	imqpro.hpp
	ImqQueue	imqque.hpp
MQBO, MQCNO, MQCSP	Gestionnaire ImqQueue	imqmgr.hpp
MQRMH	En-tête ImqReference	imqrfh.hpp
	ImqString	imqstr.hpp
MQTM	ImqTrigger	imqtrg.hpp
MQTMC		
MQTMC2	ImqTrigger	imqtrg.hpp
MQXQH		
MQWIH	En-tête ImqWork	imqwih.hpp

## ImqAuthentication-Référence croisée d'enregistrement

Référence croisée des attributs, des structures de données, des zones et des appels pour la classe C++ d'enregistrement ImqAuthentication.

*Tableau 846. Attributs, structures de données, zones et appels*

Attribut	Structure de données	Zone	Appel
nom de connexion	MQAIR	AuthInfoConnName	MQCONN
Mot de passe	MQAIR	LDAPPassword	MQCONN
type	MQAIR	AuthInfoType	MQCONN
Nom d'utilisateur	MQAIR	LDAPUserNamePtr	MQCONN
	MQAIR	LDAPUserNameDécalage	MQCONN
	MQAIR	LDAPUserNameLongueur	MQCONN

## Référence croisée ImqCache

Référence croisée des attributs et des appels pour la classe C++ ImqCache .

*Tableau 847. Attributs et appels*

Attribut	Appel
mémoire tampon automatique	MQGET
capacité de la mémoire tampon	MQGET
pointeur de mémoire tampon	MQGET, MQPUT
Longueur de données	MQGET
Position des données	MQGET
pointeur de données	MQGET
LONGUEUR DE MESSAGE	MQGET, MQPUT

## Référence croisée ImqChannel

Référence croisée des attributs, des structures de données, des zones et des appels pour la classe C++ ImqChannel .

*Tableau 848. Attributs, structures de données, zones et appels*

Attribut	Structure de données	Zone	Appel
signal de présence par lots	MQCD	BatchHeartbeat	MQCONN
nom de canal	MQCD	ChannelName	MQCONN
nom de connexion	MQCD	ConnectionName	MQCONN
	MQCD	ShortConnectionNom	MQCONN
Compression d'en-tête	MQCD	Liste HdrComp	MQCONN
intervalle des signaux de présence	MQCD	HeartbeatInterval	MQCONN
intervalle d'attente	MQCD	KeepAliveInterval	MQCONN



Tableau 848. Attributs, structures de données, zones et appels (suite)

Attribut	Structure de données	Zone	Appel
adresse locale	MQCD	LocalAddress	MQCONN
longueur maximale des messages	MQCD	Longueur maximale des messages	MQCONN
Compression de message	MQCD	Liste MsgComp	MQCONN
nom de mode	MQCD	ModeName	MQCONN
Mot de passe	MQCD	Mot de passe	MQCONN
nombre de sorties de réception	MQCD		MQCONN
noms d'exit de réception	MQCD	ReceiveExit	MQCONN
	MQCD	ReceiveExitsdéfini	MQCONN
	MQCD	ReceiveExitPtr	MQCONN
recevoir des données utilisateur	MQCD	ReceiveUserData	MQCONN
	MQCD	ReceiveUserDataPtr	MQCONN
Nom d'exit de sécurité	MQCD	SecurityExit	MQCONN
données utilisateur de sécurité	MQCD	Données SecurityUser	MQCONN
Nombre d'exits d'envoi	MQCD		MQCONN
noms d'exit d'émission	MQCD	SendExit	MQCONN
	MQCD	SendExitsdéfini	MQCONN
	MQCD	SendExitPtr	MQCONN
envoyer des données utilisateur	MQCD	SendUserData	MQCONN
	MQCD	SendUserDataPtr	MQCONN
SSL CipherSpec	MQCD	Spécification sslCipher	MQCONN
Type d'authentification du client SSL	MQCD	Authentification sslClient	MQCONN
Nom de l'homologue SSL	MQCD	SSLPEERNAME	MQCONN
Nom de programme transactionnel	MQCD	TpName	MQCONN
type de transfert	MQCD	TransportType	MQCONN
ID utilisateur	MQCD	UserIdentifier	MQCONN

### ImqCICSBridgeHeader -Référence croisée

Référence croisée des attributs, des structures de données et des zones pour la classe C++ ImqCICSBridgeHeader .

Tableau 849. Mappage d'attributs, de structures de données et de zones

Attribut	Structure de données	Zone
code de fin anormale de pont	MQCIH	AbendCode
Descripteur ADS	MQCIH	AdsDescriptor

Tableau 849. Mappage d'attributs, de structures de données et de zones (suite)

Attribut	Structure de données	Zone
caractère de signalisation	MQCIH	AttentionId
authentificateur	MQCIH	Authentificateur
code achèvement de pont	MQCIH	Code BridgeCompletion
décalage d'erreur de pont	MQCIH	ErrorOffset
code anomalie de pont	MQCIH	BridgeReason
code d'annulation de pont	MQCIH	CancelCode
tâche conversationnelle	MQCIH	ConversationalTask
position de curseur	MQCIH	CursorPosition
jeton de fonction	MQCIH	Installation
durée de conservation de l'installation	MQCIH	Heure FacilityKeep
fonction telle que	MQCIH	FacilityLike
fonction	MQCIH	Fonction
intervalle d'attente d'obtention	MQCIH	Intervalle GetWait
type de lien	MQCIH	LinkType
identificateur de transaction suivant	MQCIH	ID NextTransaction
longueur des données de sortie	MQCIH	OutputDataLongueur
format de réponse	MQCIH	ReplyToFormat
code retour de pont	MQCIH	ReturnCode
code de démarrage	MQCIH	StartCode
statut de fin de tâche	MQCIH	Statut TaskEnd
ID transaction	MQCIH	TransactionId
contrôle d'unité de travail	MQCIH	UowControl
version	MQCIH	Version

### ImqDeadLetterHeader référence croisée

Référence croisée des attributs, des structures de données et des zones pour la classe C++ ImqDeadLetterHeader .

Tableau 850. Mappage d'attributs, de structures de données et de zones

Attribut	Structure de données	Zone
code raison de rebut	MQDLH	Motif
Nom du gestionnaire de files d'attente de destination	MQDLH	DestQMgrName
nom file d'attente de destination	MQDLH	DestQName
Nom d'application d'insertion	MQDLH	PutApplName

Tableau 850. Mappage d'attributs, de structures de données et de zones (suite)

Attribut	Structure de données	Zone
Type d'application d'insertion	MQDLH	PutApplType
Date d'insertion	MQDLH	PutDate
Heure d'insertion	MQDLH	PutTime

### Référence croisée ImqError

Référence croisée des attributs et des appels pour la classe C++ ImqError .

Tableau 851. Attributs et appels

Attribut	Appel
code achèvement	MQBACK, MQBEGIN, MQCLOSE, MQCMIT, MQCONN, MQCONNX, MQDISC, MQGET, MQINQ, MQOPEN, MQPUT, MQSET
code anomalie	MQBACK, MQBEGIN, MQCLOSE, MQCMIT, MQCONN, MQCONNX, MQDISC, MQGET, MQINQ, MQOPEN, MQPUT, MQSET

### Référence croisée ImqGetMessageOptions

Référence croisée des attributs, des structures de données et des zones pour la classe C++ ImqGetMessageOptions .

Tableau 852. Mappage d'attributs, de structures de données et de zones

Attribut	Structure de données	Zone
statut de groupe	MQGMO	GroupStatus
options de mise en correspondance	MQGMO	MatchOptions
jeton de message	MQGMO	MessageToken
options	MQGMO	Options
Nom de file d'attente résolu	MQGMO	ResolvedQName
longueur renvoyée	MQGMO	ReturnedLength
segmentation	MQGMO	Segmentation
statut de segment	MQGMO	SegmentStatus
	MQGMO	Signal1
	MQGMO	Signal2
participation au point de synchronisation	MQGMO	Options
intervalle d'attente	MQGMO	WaitInterval

### Référence croisée ImqHeader

Référence croisée des attributs, des structures de données et des zones pour la classe C++ ImqHeader .

Tableau 853. Mappage d'attributs, de structures de données et de zones

Attribut	Structure de données	Zone
jeu de caractères	MQDLH, MQIIH	CodedCharSetId
codage	MQDLH, MQIIH	Codage
format	MQDLH, MQIIH	Format
indicateurs d'en-tête	MQIIH, MQRMH	Indicateurs

### ImqIMSBridgeHeader -Référence croisée

Référence croisée des attributs, des structures de données et des zones pour la classe C++ d'enregistrement ImqAuthentication.

Tableau 854. Mappage d'attributs, de structures de données et de zones

Attribut	Structure de données	Zone
authentificateur	MQIIH	Authentificateur
Mode de validation	MQIIH	CommitMode
Substitution de terminal logique	MQIIH	LTermOverride
Nom de mappe des services de structuration de messages	MQIIH	MFSMapName
format de réponse	MQIIH	ReplyToFormat
portée de sécurité	MQIIH	SecurityScope
ID instance de transaction	MQIIH	ID TranInstance
ETAT DE TRANSACTION	MQIIH	TranState

### Référence croisée ImqItem

Référence croisée des attributs et des appels pour la classe C++ ImqItem .

Tableau 855. Attributs et appels

Attribut	Appel
ID structure	MQGET

### Référence croisée ImqMessage

Référence croisée des attributs, des structures de données, des zones et des appels pour la classe C++ ImqMessage .

Tableau 856. Attributs, structures de données, zones et appels

Attribut	Structure de données	Zone	Appel
Données d'ID application	MQMD	ApplIdentityData	
Données d'origine de l'application	MQMD	ApplOriginData	
nombre d'annulations	MQMD	BackoutCount	
jeu de caractères	MQMD	CodedCharSetId	

Tableau 856. Attributs, structures de données, zones et appels (suite)

Attribut	Structure de données	Zone	Appel
codage	MQMD	Codage	
expiration	MQMD	Expiration	
format	MQMD	Format	
Indicateurs de message	MQMD	MsgFlags	
type de message	MQMD	MsgType	
décalage	MQMD	Décalage	
Longueur d'origine	MQMD	OriginalLength	
persistance	MQMD	Persistance	
priority	MQMD	Priorité	
Nom d'application d'insertion	MQMD	PutApplName	
Type d'application d'insertion	MQMD	PutApplType	
Date d'insertion	MQMD	PutDate	
Heure d'insertion	MQMD	PutTime	
nom du gestionnaire de files d'attente de réponse	MQMD	ReplyToQMgr	
Nom de file d'attente de réponses	MQMD	ReplyToQ	
rapport	MQMD	Rapport	
numéro de séquence	MQMD	MsgSeqNumber	
longueur totale des messages		DataLength	MQGET
ID utilisateur	MQMD	UserIdentifier	

### Référence croisée de la fonction de suivi ImqMessage

Référence croisée des attributs, des structures de données et des zones pour la classe C++ du dispositif de suivi ImqMessage.

Tableau 857. Mappage d'attributs, de structures de données et de zones

Attribut	Structure de données	Zone
jeton de comptabilité	MQMD	AccountingToken
ID de corrélation	MQMD	CorrelId
commentaire	MQMD	Commentaires
ID de groupe	MQMD	GroupId
ID message	MQMD	MsgId

### Référence croisée ImqNamelist

Référence croisée des attributs, des demandes et des appels pour la classe C++ ImqNamelist .

Tableau 858. Attributs, demandes et appels

Attribut	Requête	Appel
Nombre de noms	NOM_NOM_NOM_MQT	MQINQ
nom de liste de noms	NOM_NOM_MQCA	MQINQ

## Référence croisée ImqObject

Référence croisée des attributs, des structures de données, des zones, des demandes et des appels pour la classe C++ ImqObject .

Tableau 859. Attributs, structures de données, zones, demandes et appels

Attribut	Structure de données	Zone	Requête	Appel
Date de modification			MQCA_ALTERATION_DATE	MQINQ
Heure de modification			MQCA_ALTERATION_TIME	MQINQ
Autre ID utilisateur	MQOD	AlternateUserid		
Autre ID de sécurité				
options de fermeture				MQCLOSE
description			MQCA_Q_DESC, MQCA_Q_MGR_DESC, MQCA_PROCESS_DESC	MQINQ
Nom	MQOD	ObjectName	MQCA_Q_MGR_NAME, MQCQ_Q_NAME, MQCA_PROCESS_NAME	MQINQ
Options d'ouverture				MQOPEN
état ouvert				MQOPEN, MQCLOSE
Identificateur du gestionnaire de files	Identificateur du gestionnaire de files		MQCA_Q_MGR_IDENTIFIER	MQINQ

## Référence croisée ImqProcess

Référence croisée des attributs, des demandes et des appels pour la classe C++ d'enregistrement ImqAuthentication.

Tableau 860. Attributs, demandes et appels

Attribut	Requête	Appel
ID application	ID_APPL_MQCA	MQINQ
type d'application	TYPE_APPL_MQAI	MQINQ
données d'environnement	DONNEES_ENV_MQCA	MQINQ
données utilisateur	DONNEES_UTILISATEUR_MQCA	MQINQ

## Référence croisée ImqPutMessageOptions

Référence croisée des attributs, des structures de données et des zones pour la classe C++ d'enregistrement ImqAuthentication.

*Tableau 861. Mappage d'attributs, de structures de données et de zones*

Attribut	Structure de données	Zone
référence de contexte	MQPMO	Contexte
	MQPMO	InvalidDestCount
	MQPMO	KnownDestCount
options	MQPMO	Options
zones d'enregistrement	MQPMO	PutMsgRecFields
Nom de gestionnaire de files d'attente résolu	MQPMO	ResolvedQMgrNom
Nom de file d'attente résolu	MQPMO	ResolvedQName
	MQPMO	Délai d'attente
	MQPMO	UnknownDestCount
participation au point de synchronisation	MQPMO	Options

## Référence croisée ImqQueue

Référence croisée des attributs, des structures de données, des zones, des demandes et des appels pour la classe C++ ImqQueue .

*Tableau 862. Référence croisée ImqQueue*

Attribut	Structure de données	Zone	Requête	Appel
Nom de file d'attente d'annulation			MQCA_BACKOUT_REQ_Q_NAME	MQINQ
seuil d'annulation			MQIA_BACKOUT_SEUIL	MQINQ
nom file d'attente de base			MQCA_BASE_Q_NAME	MQINQ
nom de cluster			NOM_CLUSTER_MQCA_	MQINQ
Liste de noms du cluster			MQCA_CLUSTER_NAMELIST	MQINQ
Rang charge cluster			MQIA_CLWL_Q_RANK	MQINQ
Priorité charge cluster			MQIA_CLWL_Q_PRIORITY	MQINQ
File d'attente d'utilisation de charge de travail de cluster			MQIA_CLWL_USEQ	MQINQ
date de création			DATE DE CREATION_MQCA	MQINQ
heure de création			HEURE DE CREATION_MQCA	MQINQ
Longueur en cours			MQIA_CURRENT_Q_DEPTH	MQINQ

Tableau 862. Référence croisée ImqQueue (suite)

Attribut	Structure de données	Zone	Requête	Appel
liaison par défaut			MQIA_BIND par défaut	MQINQ
Option d'ouverture en entrée par défaut			MQIA_DEF_INPUT_OPEN_OPTION	MQINQ
Persistance par défaut			MQIA_DEF_PERSISTENCE	MQINQ
Priorité par défaut			MQIA_DEF_PRIORITE	MQINQ
type de définition			MQIA_DEFINITION_TYPE	MQINQ
événement de profondeur élevée			MQIA_Q_DEPTH_HIGH_EVENT	MQINQ
limite supérieure de la profondeur			MQIA_Q_DEPTH_HIGH_LIMIT	MQINQ
événement de profondeur faible			MQIA_Q_DEPTH_LOW_EVENT	MQINQ
limite inférieure de la profondeur			MQIA_Q_DEPTH_LOW_LIMIT	MQINQ
Événement de profondeur maximale			MQIA_Q_DEPTH_MAX_LIMIT	MQINQ
listes de distribution			MQIA_DIST_LISTS	MQINQ, MQSET
Nom de la file d'attente dynamique	MQOD	DynamicQName		
Sauvegarde nb d'annulations			MQIA_HARDEN_GET_BACKOUT	MQINQ
Type d'index			TYPE_INDEX_MQ	MQINQ
empêcher l'obtention			MQIA_INHIBIT_GET	MQINQ, MQSET
interdire l'insertion			MQIA_INHIBIT_PUT	MQINQ, MQSET
Nom de la file d'attente d'initialisation			NOM_Q_INITIATION_MQCA_	MQINQ
profondeur maximale			MQIA_MAX_Q_DEPTH	MQINQ
longueur maximale des messages			LONGUEUR_MSG_MAX_MQIA_MAX	MQINQ
Séquence livraison messages			MQIA_MSG_DELIVERY_SEQUENCE	MQINQ
file d'attente répartie suivante				
Classe msgs non persistants			CLASSE_NPM_MQIA_	MQINQ



Tableau 862. Référence croisée ImqQueue (suite)

Attribut	Structure de données	Zone	Requête	Appel
Nb d'ouvertures en entrée			MQIA_NOMBRE_ENTRÉES_OUVERTE	MQINQ
Nb d'ouvertures en sortie			MQIA_NOMBRE_SORTIES_OUVERTES	MQINQ
file d'attente distribuée précédente				
nom de processus			MQCA_NOM_PROCESSUS	MQINQ
Comptabilité file			MQIA_ACCOUNTING_Q	MQINQ
nom de gestionnaire de files d'attente	MQOD	ObjectQMgrName		
Ctrl file d'attente			MQIA_MONITORING_Q	MQINQ
statistiques de file d'attente			MQIA_STATISTICS_Q	MQINQ
type de file d'attente			MQIA_Q_TYPE	MQINQ
Nom du gestionnaire de files d'attente éloignées			MQCA_REMOTE_Q_MGR_NAME	MQINQ
Nom de la file d'attente distante			MQCA_NOM_QTE_DISTANT	MQINQ
Nom de gestionnaire de files d'attente résolu	MQOD	ResolvedQMgrNom		
Nom de file d'attente résolu	MQOD	ResolvedQName		
intervalle de rétention			INTERVALLE_CONSERVATION-MQIA_RETENTION_INTERVALLE	MQINQ
portée			PORT_MQAI	MQINQ
intervalle de maintenance			MQIA_Q_SERVICE_INTERVAL	MQINQ
événement d'intervalle de maintenance			MQIA_Q_ÉVÉNEMENT_INTERVALLE_SERVICE	MQINQ
Partage de file d'attente			MQIA_SHAREABILITY	MQINQ
classe d'archivage			MQCA_STORAGE_CLASS	MQINQ
Nom de la file d'attente de transmission			MQCA_XMIT_Q_NAME	MQINQ
Contrôle du déclenchement			MQIA_TRIGGER_CONTROL	MQINQ, MQSET

Tableau 862. Référence croisée ImqQueue (suite)

Attribut	Structure de données	Zone	Requête	Appel
données de déclenchement			DONNEES_TRIGGER_MQCA	MQINQ, MQSET
Longueur de déclenchement			PROFONDEUR-MQIA_TRIGGER_PROFONDEUR	MQINQ, MQSET
Priorité msg pour déclench.			MQIA_TRIGGER_MSG_PRIORITY	MQINQ, MQSET
type de déclencheur			TYPE_TRIGGER_MQIA_	MQINQ, MQSET
utilisation			MQIA_UTILISATION	MQINQ

### Références croisées du gestionnaire ImqQueue

Référence croisée des attributs, des structures de données, des zones, des demandes et des appels pour la classe C++ du gestionnaire ImqQueue.

Tableau 863. Attributs, structures de données, zones, demandes et appels

Attribut	Structure de données	Zone	Requête	Appel
remplacement des connexions de comptabilité			MQIA_ACCOUNTING_CONN_OVERRIDE	MQINQ
intervalle de comptabilité			MQIA_ACCOUNTING_INTERVAL	MQINQ
enregistrement de l'activité			ENREGISTREMENT_ACTIVITÉ_MQIA_ENREGISTREMENT	MQINQ
Adoption de la vérification du nouveau MCA			MQIA_ADOPTNEWMCA_CHECK	MQINQ
Adoption du nouveau type de MCA			MQIA_ADOPTNEWMCA_TYPE	MQINQ
type d'authentification	MQCSP	AuthenticationType		MQCONN
événement d'autorité			MQIA_AUTHORITY_EVENT	MQINQ
Options de début	Objet MQBO	Options		MQBEGIN
événement de pont			MQIA_ÉVÉNEMENT_PONT	MQINQ
Définition automatique de canal			MQIA_CHANNEL_AUTO_DEF	MQINQ

Tableau 863. Attributs, structures de données, zones, demandes et appels (suite)

Attribut	Structure de données	Zone	Requête	Appel
événement de définition automatique de canal			MQIA_CHANNEL_AUTO_EVENT	Interface MQIA
Exit de définition automatique de canal			MQIA_CHANNEL_AUTO_EXIT	Interface MQIA
événement Canal			MQIA_CHANNEL_EVENT	MQINQ
Adaptateurs d'initiateur de canal			MQIA_CHINIT_ADAPTERS	MQINQ
Ctrl init. canal			MQIA_CHINIT_CONTROL	MQINQ
Répartiteurs d'initiateur de canal			MQIA_CHINIT RÉPARTITEURS	MQINQ
Démarrage automatique de la trace de l'initiateur de canal			MQIA_CHINIT_TRACE_AUTO_START	MQINQ
Taille de la table de trace de l'initiateur de canal			MQIA_CHINIT_TRACE_TABLE_SIZE	MQINQ
Contrôle des canaux			MQIA_CANAL_SURVEILLANCE	MQINQ
référence de canal	MQCD	ChannelType		MQCONN
statistiques de canal			MQIA_STATISTICS_CANAL	MQINQ
jeu de caractères			MQIA_CODED_CHAR_SET_ID	MQINQ
Ctrl émetteur cluster			MQIA_MONITORING_AUTO_CLUSSDR	MQINQ
Stats émetteur cluster			MQIA_STATISTICS_AUTO_CLUSSDR	MQINQ
Données de charge de travail de cluster			MQCA_CLUSTER_DONNÉES_CHARGE_TRAVAIL_CLUSTER	MQINQ
Exit de charge de travail du cluster			MQCA_CLUSTER_WORKLOAD_EXIT	MQINQ
Longueur de charge de travail de cluster			MQIA_CLUSTER_LONGUEUR_TRAVAIL_CLUSTER	MQINQ

Tableau 863. Attributs, structures de données, zones, demandes et appels (suite)

Attribut	Structure de données	Zone	Requête	Appel
mru de charge de travail de cluster			MQIA_CLWL_MRU_CHANNELS	MQINQ
File d'attente d'utilisation de charge de travail de cluster			MQIA_CLWL_USEQ	MQINQ
événement de commande			ÉVÉNEMENT_COMMANDE_MQIA_COMMANDE	MQINQ
Nom de la file d'attente d'entrée des commandes			MQCA_COMMAND_INPUT_Q_NAME	MQINQ
niveau de commande			MQIA_NIVEAU DE COMMANDE	MQINQ
Contrôle du serveur de commandes			MQIA_CMD_SERVER_CONTROL	MQINQ
Options de connexion	MQCNO	Options		MQCONN, MQCONNX
ID de connexion	MQCNO	ConnectionId		MQCONNX
état de connexion				MQCONN, MQCONNX, MQDISC
balise de connexion	MQCD	ConnTag		MQCONNX
matériel de cryptographie	MQSCO	CryptoHardware		MQCONNX
Nom de la file d'attente de rebut			MQCA_NOM_LETTRE_INTERBLOCAGE	MQINQ
nom de file de transmission par défaut			MQCA_DEF_XMIT_Q_NAME	MQINQ
listes de distribution			MQIA_DIST_LISTS	MQINQ
groupe dns			GROUPE_DNS_MQCA	MQINQ
dns wlm			MQIA_DNS_WLM	MQINQ
premier enregistrement d'authentification	MQSCO	AuthInfoRecOffset		MQCONNX
	MQSCO	AuthInfoRecPtr		MQCONNX

Tableau 863. Attributs, structures de données, zones, demandes et appels (suite)

Attribut	Structure de données	Zone	Requête	Appel
événement d'interdiction			MQIA_INHIBIT_EVENT	MQINQ
Version adresse IP			MQIA_IP_ADRESSE_VERSION	MQINQ
référentiel de clés	MQSCO	KeyRepository		MQCONN
nombre de réinitialisations de clé	MQSCO	Nombre de KeyReset		MQCONN
Délai du programme d'écoute			MQIA_LISTENER_TIMER	MQINQ
événement local			MQIA_LOCAL_EVENT	MQINQ
événement de consignateur			MQIA_LOGGER_EVENT	MQINQ
Nom de groupe de LU			NOM_GROUPE_LU_MQCA	MQINQ
Nom de LU			NOM_LU_MQCA	MQINQ
Suffixe de bras lu62			MQCA_LU62_ARM_SUFFIX	MQINQ
Canaux lu62			MQIA_LU62_CHANNELS	MQINQ
nombre maximal de canaux actifs			MQIA_ACTIVE_CHANNELS	MQINQ
Nombre maximal de canaux			CANAL_MAX_MQIA_CANAL	MQINQ
nombre maximal de descripteurs			MQIA_MAX_DESCRIPTEURS	MQINQ
longueur maximale des messages			LONGUEUR_MSG_MAX_MQIA_MAX	MQINQ
Priorité maximum			MQIA_MAX_PRIORITE	MQINQ
Nb max. messages non validés			MQIA_MAX_UNCOMMITTED_MSGS	MQINQ
Comptabilité MQI			MQIA_ACCOUNTING_MQI	MQINQ
Statistiques MQI			MQIA_STATISTICS_MQI	MQINQ
port de communications sortantes maximum			MQIA_OUTBOUND_PORT_MAX	MQINQ

Tableau 863. Attributs, structures de données, zones, demandes et appels (suite)

Attribut	Structure de données	Zone	Requête	Appel
port de communications sortantes minimum			MQIA_OUTBOUND_PORT_MIN	MQINQ
Mot de passe	MQCSP	CSPPasswordPtr		MQCONN
	MQCSP	CSPPasswordOffset		MQCONN
	MQCSP	CSPPasswordLength		MQCONN
événement Performances			ÉVÉNEMENT_PERFORMANCE_MQI	MQINQ
plateforme			MQIA_PLATEFORME	MQINQ
Comptabilité file			MQIA_ACCOUNTING_Q	MQINQ
Ctrl file d'attente			MQIA_MONITORING_Q	MQINQ
statistiques de file d'attente			MQIA_STATISTICS_Q	MQINQ
délai d'attente de réception			DÉLAI_RÉCEPTION_MQIA_EXPIRATION	MQINQ
délai d'attente minimal de réception			MQIA_RECEIVE_TIMEOUT_MIN	MQINQ
Type de délai de réception			MQIA_RECEIVE_TIMEOUT_TYPE	MQINQ
événement distant			ÉVÉNEMENT_MQIA_REMOTE_EVENT	MQINQ
nom de référentiel			MQCA_NOM RÉFÉRENTIEL	MQINQ
Liste de noms du référentiel			MQCA_REPOSITORY_NAMELIST	MQINQ
nom du gestionnaire de files d'attente partagées			MQIA_SHARED_Q_Q_MGR_NAME	MQINQ
événement ssl			MQIA_SSL_EVENT	MQINQ
Fips ssl			MQIA_SSL_FIPS_REQUIRED	MQINQ
Nbre avant réinit clé SSL			MQIA_SSL_RESET_COUNT	MQINQ
événement de démarrage / arrêt			MQIA_START_STOP_EVENT	MQINQ
intervalle de collecte de statistiques			INTERVALLE_MQIA_STATISTICS_INTERVAL	MQINQ

Tableau 863. Attributs, structures de données, zones, demandes et appels (suite)

Attribut	Structure de données	Zone	Requête	Appel
Prise en charge points sync.			MQIA_SYNCPOINT	MQINQ
canaux tcp			MQIA_TCP_CHANNELS	MQINQ
Signal de présence TCP			MQIA_TCP_KEEP_ALIVE	MQINQ
Nom TCP			NOM_TCP_MQCA	MQINQ
Type de pile TCP			MQIA_TCP_STACK_TYPE	MQINQ
Enregistrement trace route			MQIA_TRACE_ROUTE_ENREGISTREMENT	MQINQ
Intervalle de déclenchement			INTERVALLE_DÉCLENCHEUR_MQIA_	MQINQ
ID utilisateur	MQCSP	CSPUserIdPtr		MQCONN
	MQCSP	CSPUserIdDécalage		MQCONN
	MQCSP	CSPUserIdLongueur		MQCONN

### Référence croisée d'en-tête ImqReference

Référence croisée des attributs, des structures de données et des zones pour la classe C++ d'enregistrement ImqAuthentication.

Tableau 864. Mappage d'attributs, de structures de données et de zones

Attribut	Structure de données	Zone
environnement de destination	MQRMH	DestEnvLongueur, DestEnvDécalage
nom de destination	MQRMH	DestNameLongueur, DestNameDécalage
id d'instance	MQRMH	ID ObjectInstance
longueur logique	MQRMH	Longueur DataLogical
décalage logique	MQRMH	DataLogical
décalage logique 2	MQRMH	DataLogicalOffset2
type de référence	MQRMH	ObjectType
environnement source	MQRMH	SrcEnvLongueur, SrcEnvDécalage
nom de source	MQRMH	SrcNameLongueur, SrcNameDécalage

### Référence croisée ImqTrigger

Référence croisée des attributs, des structures de données et des zones pour la classe C++ d'enregistrement ImqAuthentication.

Tableau 865. Mappage d'attributs, de structures de données et de zones

Attribut	Structure de données	Zone
ID application	MQTM	ApplId
type d'application	MQTM	ApplType
données d'environnement	MQTM	EnvData
nom de processus	MQTM	ProcessName
nom de la file d'attente	MQTM	QName
données de déclenchement	MQTM	TriggerData
données utilisateur	MQTM	UserData

### Référence croisée d'en-tête ImqWork

Référence croisée des attributs, des structures de données et des zones pour la classe C++ d'enregistrement ImqAuthentication.

Tableau 866. Mappage d'attributs, de structures de données et de zones

Attribut	Structure de données	Zone
jeton de message	MQWIH	MessageToken
nom de service	MQWIH	ServiceName
étape de service	MQWIH	ServiceStep

### Classe C++ d'enregistrement ImqAuthentication

Cette classe encapsule un enregistrement d'informations d'authentification (MQAIR) à utiliser lors de l'exécution de la méthode ImqQueueManager: :connect, pour les connexions client TLS personnalisées.

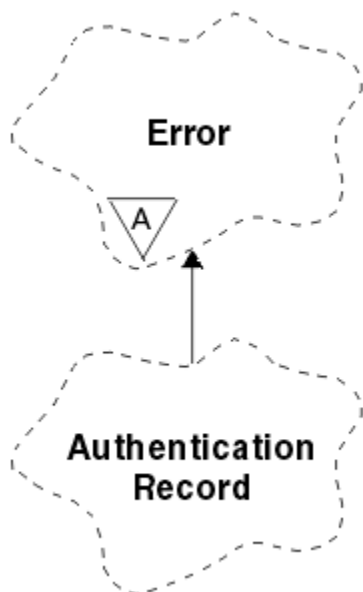


Figure 14. Classe d'enregistrement ImqAuthentication



Pour plus de détails, voir la description de la méthode `ImqQueueManager::connect`. Cette classe n'est pas disponible sur la plateforme z/OS .

- «Attributs d'objet», à la page [1889](#)
- «Constructeurs», à la page [1889](#)
- «Méthodes d'objet (public)», à la page [1889](#)
- «Méthodes d'objet (protégées)», à la page [1890](#)

## Attributs d'objet

### nom de connexion

Nom de la connexion au serveur CRL LDAP. Il s'agit de l'adresse IP ou du nom DNS, suivi éventuellement du numéro de port, entre parenthèses.

### référence de connexion

Référence à un objet gestionnaire `ImqQueue` qui fournit la connexion requise à un gestionnaire de files d'attente (local). La valeur initiale est zéro. Ne le confondez pas avec le nom du gestionnaire de files d'attente qui identifie un gestionnaire de files d'attente (éventuellement distant) pour une file d'attente nommée.

### enregistrement d'authentification suivant

Objet suivant de cette classe, sans ordre particulier, ayant la même **référence de connexion** que cet objet. La valeur initiale est zéro.

### Mot de passe

Mot de passe fourni pour l'authentification de connexion au serveur CRL LDAP.

### enregistrement d'authentification précédent

Objet précédent de cette classe, sans ordre particulier, ayant la même **référence de connexion** que cet objet. La valeur initiale est zéro.

### type

Type des informations d'authentification contenues dans l'enregistrement.

### Nom d'utilisateur

Identificateur utilisateur fourni pour l'autorisation sur le serveur CRL LDAP.

## Constructeurs

### `ImqAuthenticationRecord ()` ;

Constructeur par défaut.

## Méthodes d'objet (public)

### `void operator = (const ImqAuthenticationRecord & air) ;`

Copie les données d'instance à partir de *air*, en remplaçant les données d'instance existantes.

### `const ImqString & connectionName () const ;`

Renvoie le **nom de connexion**.

### `void setConnectionName (const ImqString & name) ;`

Définit le **nom de connexion**.

### `void setConnectionName (const char * name = 0) ;`

Définit le **nom de connexion**.

### `ImqQueueManager * connectionReference () const ;`

Renvoie la **référence de connexion**.

### `void setConnectionReference ( ImqQueueManager & manager ) ;`

Définit la **référence de connexion**.

### `void setConnectionReference ( ImqQueueManager * manager = 0) ;`

Définit la **référence de connexion**.

**void copyOut (MQAIR \* pAir) ;**

Copie les données d'instance dans *pAir*, en remplaçant les données d'instance existantes. Cela peut impliquer l'allocation de mémoire dépendante.

**void clear (MQAIR \* pAir) ;**

Efface la structure et libère le stockage dépendant référencé par *pAir*.

**ImqAuthenticationEnregistrement \* nextAuthenticationEnregistrement () const ;**

Renvoie l' **enregistrement d'authentification suivant**.

**const ImqString & password () const ;**

Renvoie le **mot de passe**.

**void setPassword (const ImqString & mot\_de\_passe) ;**

Définit le **mot de passe**.

**void setPassword (const char \* password = 0) ;**

Définit le **mot de passe**.

**ImqAuthenticationRecord \* previousAuthenticationRecord () const ;**

Renvoie l' **enregistrement d'authentification précédent**.

**Type MQLONG () const ;**

Renvoie le **type**.

**void setType (const MQLONG type) ;**

Définit le **type**.

**const ImqString & userName () const ;**

Renvoie le **nom d'utilisateur**.

**void setUsername (const ImqString & name) ;**

Définit le **nom d'utilisateur**.

**void setUsername (const char \* name = 0) ;**

Définit le **nom d'utilisateur**.

### Méthodes d'objet (protégées)

**void setNextAuthenticationRecord ( ImqAuthenticationRecord \* pAir = 0) ;**

Définit l' **enregistrement d'authentification suivant**.

**Attention:** utilisez cette fonction uniquement si vous êtes certain qu'elle ne rompt pas la liste des enregistrements d'authentification.

**void setPreviousAuthenticationRecord ( ImqAuthenticationRecord \* pAir = 0) ;**

Définit l' **enregistrement d'authentification précédent**.

**Attention:** utilisez cette fonction uniquement si vous êtes certain qu'elle ne rompt pas la liste des enregistrements d'authentification.

## Classe C++ ImqBinary

Cette classe encapsule un tableau d'octets binaire qui peut être utilisé pour les valeurs ImqMessage **jeton de comptabilité**, **ID de corrélation** et **ID de message** . Il facilite l'affectation, la copie et la comparaison.

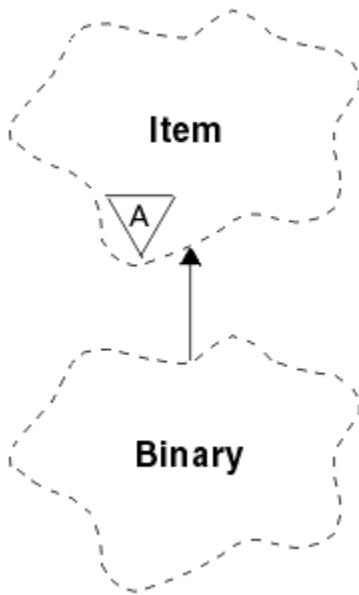


Figure 15. Classe *ImqBinary*

- [«Attributs d'objet»](#), à la page 1891
- [«Constructeurs»](#), à la page 1891
- [«Méthodes \*ImqItem\* surchargées»](#), à la page 1891
- [«Méthodes d'objet \(public\)»](#), à la page 1892
- [«Méthodes d'objet \(protégées\)»](#), à la page 1892
- [«Codes raison»](#), à la page 1892

## Attributs d'objet

### données

Tableau d'octets de données binaires. La valeur initiale est null.

### Longueur de données

Nombre d'octets. La valeur initiale est zéro.

### pointeur de données

Adresse du premier octet des **données**. La valeur initiale est zéro.

## Constructeurs

### **ImqBinary();**

Constructeur par défaut.

### **ImqBinary( const ImqBinary & binaire );**

Constructeur de copie.

### **ImqBinary( const void \* data, const size\_t longueur );**

Copie *longueur* octets à partir de *données*.

## Méthodes *ImqItem* surchargées

### **virtual ImqBoolean copyOut ( ImqMessage & msg );**

Copie les **données** dans la mémoire tampon de messages, en remplaçant tout contenu existant. Définit le **msg format** sur MQFMT\_NONE.

Voir la description de la méthode de classe *ImqItem* pour plus de détails.

### **virtuel ImqBoolean pasteIn ( ImqMessage & msg ) ;**

Définit les **données** en transférant les données restantes de la mémoire tampon de messages, en remplaçant les **données** existantes.

Pour que l'opération aboutisse, le format ImqMessage doit être MQFMT\_NONE.

Voir la description de la méthode de classe ImqItem pour plus de détails.

### **Méthodes d'objet (public)**

#### **void operator = ( const ImqBinary & binaire ) ;**

Copie les octets à partir de *binaire*.

#### **ImqBoolean operator == ( const ImqBinary & binaire ) ;**

Compare cet objet à *binaire*. Elle renvoie FALSE si elle n'est pas égale et TRUE dans le cas contraire. Les objets sont égaux s'ils ont la même **longueur de données** et que les octets correspondent.

#### **ImqBoolean copyOut ( void \* buffer, const size\_t length, const char pad = 0 ) ;**

Copie jusqu'à *longueur* octets du **pointeur de données** dans la *mémoire tampon*. Si la **longueur des données** est insuffisante, l'espace restant dans la *mémoire tampon* est rempli avec *pad* octets. *buffer* peut être égal à zéro si *length* est également égal à zéro. *length* ne doit pas être négatif. Elle renvoie TRUE en cas de succès.

#### **size\_t dataLength () const ;**

Renvoie la **longueur des données**.

#### **ImqBoolean setDataLongueur ( const size\_t longueur ) ;**

Définit la **longueur des données**. Si la **longueur des données** est modifiée suite à cette méthode, les données de l'objet ne sont pas initialisées. Elle renvoie TRUE en cas de succès.

#### **void \* dataPointer () const ;**

Renvoie le **pointeur de données**.

#### **ImqBoolean isNull () const ;**

Renvoie TRUE si la **longueur des données** est égale à zéro ou si tous les octets de **données** sont égaux à zéro. Sinon, elle retourne FALSE.

#### **ImqBoolean set ( const void \* buffer, const size\_t longueur ) ;**

Copie *longueur* octets de la *mémoire tampon*. Elle renvoie TRUE en cas de succès.

### **Méthodes d'objet (protégées)**

#### **void clear () ;**

Réduit la **longueur des données** à zéro.

### **Codes raison**

- MQRC\_NO\_BUFFER
- MQRC\_STORAGE\_NOT\_AVAILABLE
- MQRC\_INCONSISTENT\_FORMAT

## **Classe C++ ImqCache**

Utilisez cette classe pour stocker ou convertir des données en mémoire.

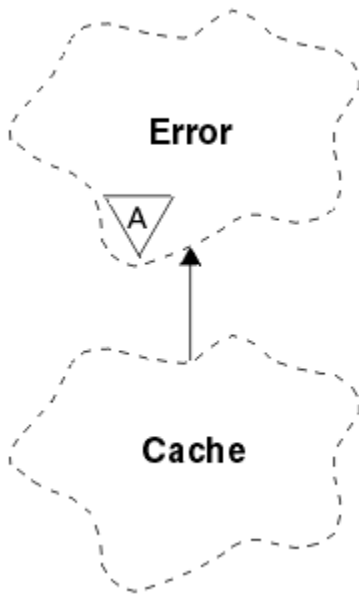


Figure 16. Classe *ImqCache*

Utilisez cette classe pour stocker ou convertir des données en mémoire. Vous pouvez désigner une mémoire tampon de taille fixe ou le système peut fournir une quantité de mémoire flexible automatiquement. Cette classe est liée aux appels MQI répertoriés dans [«Référence croisée ImqCache»](#), à la page 1872.

- [«Attributs d'objet»](#), à la page 1893
- [«Constructeurs»](#), à la page 1894
- [«Méthodes d'objet \(public\)»](#), à la page 1894
- [«Codes raison»](#), à la page 1895

## Attributs d'objet

### mémoire tampon automatique

Indique si la mémoire tampon est gérée automatiquement par le système (TRUE) ou si elle est fournie par l'utilisateur (FALSE). Il est initialement défini sur TRUE.

Cet attribut n'est pas défini directement. Il est défini indirectement à l'aide de la méthode **useEmptyBuffer** ou de la méthode **useFullBuffer** .

Si le stockage utilisateur est fourni, cet attribut est FALSE, la mémoire tampon ne peut pas augmenter et des erreurs de dépassement de mémoire tampon peuvent se produire. L'adresse et la longueur de la mémoire tampon restent constantes.

Si le stockage utilisateur n'est pas fourni, cet attribut est TRUE et la mémoire tampon peut croître de manière incrémentielle pour contenir une quantité arbitraire de données de message. Cependant, lorsque la taille de la mémoire tampon augmente, l'adresse de la mémoire tampon peut changer. Par conséquent, soyez prudent lorsque vous utilisez le **pointeur de mémoire tampon** et le **pointeur de données**.

### capacité de la mémoire tampon

Nombre d'octets de mémoire dans la mémoire tampon. La valeur initiale est zéro.

### pointeur de mémoire tampon

Adresse de la mémoire tampon. La valeur initiale est null.

### Longueur de données

Nombre d'octets suivant le **pointeur de données**. Cette valeur doit être inférieure ou égale à la **longueur du message**. La valeur initiale est zéro.

## Position des données

Nombre d'octets précédant le **pointeur de données**. Cette valeur doit être inférieure ou égale à la **longueur du message**. La valeur initiale est zéro.

## pointeur de données

Adresse de la partie de la mémoire tampon qui doit être écrite ou lue à partir de la suivante. La valeur initiale est null.

## LONGUEUR DE MESSAGE

Nombre d'octets de données significatives dans la mémoire tampon. La valeur initiale est zéro.

## Constructeurs

### ImqCache();

Constructeur par défaut.

### ImqCache( const ImqCache & cache );

Constructeur de copie.

## Méthodes d'objet (public)

### void operator = ( const ImqCache & cache );

Copie jusqu'à **longueur de message** octets de données de l'objet *cache* vers l'objet. Si la **mémoire tampon automatique** est FALSE, la **longueur de la mémoire tampon** doit déjà être suffisante pour accueillir les données copiées.

### ImqBoolean automaticBuffer () const ;

Renvoie la valeur de la **mémoire tampon automatique** .

### size\_t bufferLength () const ;

Renvoie la **longueur de la mémoire tampon**.

### char \* bufferPointer () const ;

Renvoie le **pointeur de mémoire tampon**.

### void clearMessage () ;

Définit la **longueur de message** et le **décalage de données** sur zéro.

### size\_t dataLength () const ;

Renvoie la **longueur des données**.

### size\_t dataOffset () const ;

Renvoie le **décalage de données**.

### ImqBoolean setDataDécalage ( const size\_t décalage );

Définit le **décalage de données**. La **longueur de message** est augmentée si nécessaire pour s'assurer qu'elle n'est pas inférieure au **décalage de données**. Cette méthode renvoie TRUE en cas de succès.

### char \* dataPointer () const ;

Renvoie une copie du **pointeur de données**.

### size\_t messageLength () const ;

Renvoie la **longueur de message**.

### ImqBoolean setMessageLongueur ( const size\_t longueur );

Définit la **longueur de message**. Augmente la **longueur de la mémoire tampon** si nécessaire pour que la **longueur de message** ne soit pas supérieure à la **longueur de la mémoire tampon**. Réduit le **décalage de données** si nécessaire pour s'assurer qu'il n'est pas supérieur à la **longueur de message**. Elle renvoie TRUE en cas de succès.

### ImqBoolean moreBytes ( const size\_t bytes-required );

S'assure que *bytes-required* octets supplémentaires sont disponibles (pour l'écriture) entre le **pointeur de données** et la fin de la mémoire tampon. Elle renvoie TRUE en cas de succès.

Si **automatic buffer** a la valeur TRUE, davantage de mémoire est acquise selon les besoins ; sinon, la **longueur de la mémoire tampon** doit déjà être adéquate.

**ImqBoolean read ( const size\_t longueur, char \* & external-buffer ) ;**

Copie *longueur* octets, à partir de la mémoire tampon commençant à la position **pointeur de données**, dans la *mémoire tampon externe*. Une fois les données copiées, le **décalage de données** est augmenté de *longueur*. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean resizeBuffer ( const size\_t longueur ) ;**

Fait varier la **longueur de la mémoire tampon**, à condition que la **mémoire tampon automatique** soit TRUE. Ceci est réalisé en réallouant la mémoire tampon. Jusqu'à **longueur de message** octets de données de la mémoire tampon existante sont copiés dans la nouvelle. Le nombre maximal d'octets copiés est de *longueur*. Le **pointeur de mémoire tampon** est modifié. La **longueur des messages** et le **décalage des données** sont préservés aussi près que possible dans les limites de la nouvelle mémoire tampon. Elle renvoie TRUE en cas de succès, et FALSE si la **mémoire tampon automatique** est FALSE.

**Remarque :** Cette méthode peut échouer avec MQRC\_STORAGE\_NOT\_AVAILABLE en cas de problème avec les ressources système.

**ImqBoolean useEmptyBuffer ( const char \* external-buffer, const size\_t longueur ) ;**

Identifie une mémoire tampon utilisateur vide, en définissant le **pointeur de mémoire tampon** pour qu'il pointe vers *external-buffer*, la **longueur de mémoire tampon** sur *longueur* et la **longueur de message** sur zéro. Effectue un **clearMessage**. Si la mémoire tampon est entièrement amorcée avec des données, utilisez la méthode **useFullBuffer** à la place. Si la mémoire tampon est partiellement primée avec des données, utilisez la méthode **setMessageLength** pour indiquer la quantité correcte. Cette méthode renvoie TRUE en cas de succès.

Cette méthode peut être utilisée pour identifier une quantité fixe de mémoire, comme décrit précédemment (*external-buffer* n'est pas null et *length* est différent de zéro), auquel cas **automatic buffer** est défini sur FALSE, ou elle peut être utilisée pour rétablir la mémoire flexible gérée par le système (*external-buffer* est null et *length* est zéro), auquel cas **automatic buffer** est défini sur TRUE.

**ImqBoolean useFullBuffer ( const char \* externalBuffer, const size\_t longueur ) ;**

Comme pour **useEmptyBuffer**, sauf que la **longueur de message** est définie sur *longueur*. Elle renvoie TRUE en cas de succès.

**ImqBoolean write ( const size\_t longueur, const char \* external-buffer ) ;**

Copie *length* octets, à partir de la *mémoire tampon externe*, dans la mémoire tampon à partir de la position **pointeur de données**. Une fois les données copiées, le **décalage de données** est augmenté de *longueur* et la **longueur de message** est augmentée si nécessaire pour s'assurer qu'elle n'est pas inférieure à la nouvelle valeur de **décalage de données**. Cette méthode renvoie TRUE en cas de succès.

Si **automatic buffer** a pour valeur TRUE, une quantité de mémoire adéquate est garantie ; dans le cas contraire, le **décalage de données** final ne doit pas dépasser la **longueur de la mémoire tampon**.

**Codes raison**

- MQRC\_BUFFER\_NOT\_AUTOMATIC
- MQRC\_DATA\_TRONQUÉ
- MQRC\_INSUFFISANT\_BUFFER
- MQRC\_DONNÉES\_INSUFFISANTES
- POINT\_NULL\_MQRC\_NULL
- MQRC\_STORAGE\_NOT\_AVAILABLE
- LONGUEUR\_ZÉRO\_MQRC\_

**Classe C++ ImqChannel**

Cette classe encapsule une définition de canal (MQCD) à utiliser lors de l'exécution de la méthode Manager: :connect pour les connexions client personnalisées.

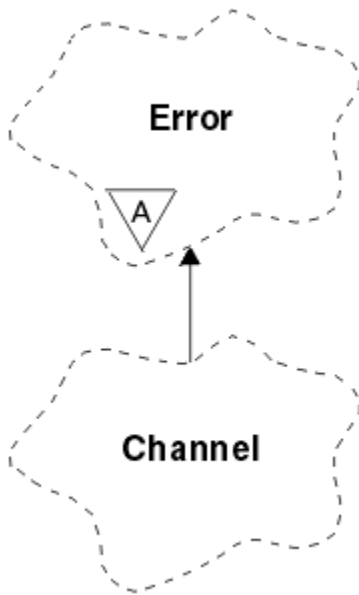


Figure 17. Classe *ImqChannel*

Pour plus de détails, voir la description de la méthode `Manager::connect` et [Exemple de programme HELLO WORLD \(imqwrlld.cpp\)](#).

Toutes les méthodes répertoriées ne sont pas applicables à toutes les plateformes. Pour plus d'informations, voir les descriptions des commandes [DEFINE CHANNEL](#) et [ALTER CHANNEL](#).

La classe `ImqChannel` n'est pas prise en charge sous z/OS.

- «Attributs d'objet», à la page 1896
- «Constructeurs», à la page 1897
- «Méthodes d'objet (public)», à la page 1897
- «Codes raison», à la page 1901

## Attributs d'objet

### signal de présence par lots

Nombre de millisecondes entre les vérifications qu'un canal distant est actif. La valeur initiale est 0.

### nom de canal

Nom du canal. La valeur initiale est null.

### nom de connexion

Nom de la connexion. Par exemple, l'adresse IP d'un ordinateur hôte. La valeur initiale est null.

### Compression d'en-tête

Liste des techniques de compression de données d'en-tête prises en charge. Les valeurs initiales sont toutes définies sur `MQCOMPRESS_NOT_AVAILABLE`.

### intervalle des signaux de présence

Nombre de secondes entre les vérifications qu'une connexion fonctionne toujours. La valeur initiale est 300.

### intervalle d'attente

Nombre de secondes passées à la pile de communications en spécifiant le délai de signal de présence pour le canal. La valeur initiale est `MQKAI_AUTO`.

### adresse locale

Adresse de communication locale du canal.



**longueur maximale des messages**

Longueur maximale de message prise en charge par le canal dans une seule communication. La valeur initiale est 4 194 304.

**Compression de message**

Liste des techniques de compression de données de message prises en charge. Les valeurs initiales sont toutes définies sur MQCOMPRESS\_NOT\_AVAILABLE.

**nom de mode**

Nom du mode. La valeur initiale est null.

**Mot de passe**

Mot de passe fourni pour l'authentification de la connexion. La valeur initiale est null.

**nombre de sorties de réception**

Nombre d'exits de réception. La valeur initiale est zéro. Cet attribut est en lecture seule.

**noms d'exit de réception**

Noms des exits de réception.

**recevoir des données utilisateur**

Données associées aux exits de réception.

**Nom d'exit de sécurité**

Nom d'un exit de sécurité à appeler côté serveur de la connexion. La valeur initiale est null.

**données utilisateur de sécurité**

Données à transmettre à l'exit de sécurité. La valeur initiale est null.

**Nombre d'exits d'envoi**

Nombre d'exits d'émission. La valeur initiale est zéro. Cet attribut est en lecture seule.

**noms d'exit d'émission**

Noms des exits d'émission.

**envoyer des données utilisateur**

Données associées aux exits d'envoi.

**SSL CipherSpec**

CipherSpec à utiliser avec TLS.

**Type d'authentification du client SSL**

Type d'authentification client à utiliser avec TLS.

**Nom de l'homologue SSL**

Nom d'homologue à utiliser avec TLS.

**Nom de programme transactionnel**

Nom du programme de transaction. La valeur initiale est null.

**type de transfert**

Type de transport de la connexion. La valeur initiale est MQXPT\_LU62.

**ID utilisateur**

Identificateur utilisateur fourni pour l'autorisation. La valeur initiale est null.

**Constructeurs****ImqChannel( ) ;**

Constructeur par défaut.

**ImqChannel( const ImqChannel & channel ) ;**

Constructeur de copie.

**Méthodes d'objet (public)****void operator = (const ImqChannel & canal) ;**

Copie les données d'instance à partir du *canal*, en remplaçant les données d'instance existantes.

**MQLONG batchHeartBeat () const ;**

Renvoie le **signal de présence du lot**.

**ImqBoolean setBatchHeartBeat(const MQLONG heartbeat = 0L ) ;**

Définit le **signal de présence du lot**. Cette méthode renvoie TRUE en cas de succès.

**ImqString channelName() const ;**

Renvoie le **nom de canal**.

**ImqBoolean setChannelNom (const char \* nom = 0) ;**

Définit le **nom de canal**. Cette méthode renvoie TRUE en cas de succès.

**ImqString connectionName() const ;**

Renvoie le **nom de connexion**.

**ImqBoolean setConnectionNom (const char \* name = 0) ;**

Définit le **nom de connexion**. Cette méthode renvoie TRUE en cas de succès.

**size\_t headerCompressionNombre () const ;**

Renvoie le nombre de techniques de compression de données d'en-tête prises en charge.

**ImqBoolean headerCompression(nombre de tailles de const, compression MQLONG [ ]) const ;**

Renvoie des copies des techniques de compression de données d'en-tête prises en charge dans **compress**. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean setHeaderCompression (nombre de tailles de const, compression MQLONG de const [ ]) ;**

Définit les techniques de compression de données d'en-tête prises en charge sur **compress**.

Définit le nombre de techniques de compression de données d'en-tête prises en charge sur **count**.

Cette méthode renvoie TRUE en cas de succès.

**Intervalle heartBeatMQLONG () const ;**

Renvoie l' **intervalle des signaux de présence**.

**ImqBoolean setHeartBeatInterval(const MQLONG interval = 300L ) ;**

Définit l' **intervalle des signaux de présence**. Cette méthode renvoie TRUE en cas de succès.

**Intervalle keepAliveMQLONG () const ;**

Renvoie l' **intervalle de signal de présence**.

**ImqBoolean setKeepAliveInterval(const MQLONG intervalle = MQKAI\_AUTO) ;**

Définit l' **intervalle de signal de présence**. Cette méthode renvoie TRUE en cas de succès.

**ImqString localAddress() const ;**

Renvoie l' **adresse locale**.

**ImqBoolean setLocalAdresse (const char \* address = 0) ;**

Définit l' **adresse locale**. Cette méthode renvoie TRUE en cas de succès.

**MQLONG maximumMessageLongueur () const ;**

Renvoie la **longueur maximale des messages**.

**ImqBoolean setMaximumMessageLength(const MQLONG length = 4194304L ) ;**

Définit la **longueur maximale des messages**. Cette méthode renvoie TRUE en cas de succès.

**size\_t messageCompressionNombre () const ;**

Renvoie le nombre de techniques de compression de données de message prises en charge.

**ImqBoolean messageCompression(nombre de tailles de const, MQLONG compress [ ]) const ;**

Renvoie des copies des techniques de compression de données de message prises en charge dans **compress**. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean setMessageCompression (const size\_t count, const MQLONG compress [ ]) ;**

Définit les techniques de compression de données de message prises en charge à compresser.

Définit le comptage des techniques de compression de données de message prises en charge.

Cette méthode renvoie TRUE en cas de succès.

**ImqString modeName() const ;**

Renvoie le **nom de mode**.

**ImqBoolean setModeNom (const char \* name = 0) ;**

Définit le **nom de mode**. Cette méthode renvoie TRUE en cas de succès.

**Mot de passe ImqString () const ;**

Renvoie le **mot de passe**.

**ImqBoolean setPassword(const char \* password = 0) ;**

Définit le **mot de passe**. Cette méthode renvoie TRUE en cas de succès.

**size\_t receiveExitNombre () const ;**

Renvoie le **nombre d'exits de réception**.

**ImqString receiveExitNom () ;**

Renvoie le premier des **noms d'exit de réception**, le cas échéant. Si le **nombre d'exits de réception** est égal à zéro, il renvoie une chaîne vide.

**ImqBoolean receiveExitNoms (const size\_t count, ImqString \* noms [ ]) ;**

Renvoie des copies des **noms d'exit de réception** dans les *noms*. Définit tous les *noms* au-delà de **nombre d'exits de réception** sur des chaînes nulles. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean setReceiveExitName(const char \* name = 0) ;**

Définit les **noms d'exit de réception** sur le *nomunique*. *name* peut être vide ou null. Définit le **nombre d'exits de réception** sur 1 ou sur zéro. Efface la **réception des données utilisateur**. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean setReceiveExitNames(const size\_t count, const char \* names [ ]) ;**

Définit les **noms d'exit de réception** sur *noms*. Les valeurs de *noms* individuelles ne doivent pas être vides ou nulles. Définit le **nombre d'exits de réception** sur *nombre*. Efface la **réception des données utilisateur**. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean setReceiveExitNames(const size\_t count, const ImqString \* names [ ]) ;**

Définit les **noms d'exit de réception** sur *noms*. Les valeurs de *noms* individuelles ne doivent pas être vides ou nulles. Définit le **nombre d'exits de réception** sur *nombre*. Efface la **réception des données utilisateur**. Cette méthode renvoie TRUE en cas de succès.

**ImqString receiveUserData () ;**

Renvoie le premier des éléments **receive user data**, le cas échéant. Si le **nombre d'exits de réception** est égal à zéro, renvoie une chaîne vide.

**ImqBoolean receiveUserDonnées (const size\_t count, ImqString \* données [ ]) ;**

Renvoie des copies des éléments **receive user data** dans *data*. Définit toutes les *données* au-delà du **nombre d'exits de réception** sur des chaînes nulles. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean setReceiveUserData(const char \* data = 0) ;**

Définit **receive user data** sur l'élément unique *data*. Si *data* n'est pas null, le **nombre d'exits de réception** doit être au moins égal à 1. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean setReceiveUserData(const size\_t count, const char \* data [ ]) ;**

Définit le paramètre **receive user data** sur *data*. *count* ne doit pas être supérieur au **nombre d'exits de réception**. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean setReceiveUserData(const size\_t count, const ImqString \* data [ ]) ;**

Définit le paramètre **receive user data** sur *data*. *count* ne doit pas être supérieur au **nombre d'exits de réception**. Cette méthode renvoie TRUE en cas de succès.

**ImqString securityExitNom () const ;**

Renvoie le **nom de l'exit de sécurité**.

**ImqBoolean setSecurityExitName(const char \* name = 0) ;**

Définit le **nom de l'exit de sécurité**. Cette méthode renvoie TRUE en cas de succès.

**ImqString securityUserDonnées () const ;**

Renvoie les **données utilisateur de sécurité**.

**ImqBoolean setSecurityUserData(const char \* data = 0) ;**

Définit les **données utilisateur de sécurité**. Cette méthode renvoie TRUE en cas de succès.

**size\_t sendExitNombre () const ;**

Renvoie le **nombre d'exits d'émission**.

**ImqString sendExitNom () ;**

Renvoie le premier des **noms d'exit d'émission**, le cas échéant. Renvoie une chaîne vide si le **nombre d'exits d'émission** est égal à zéro.

**ImqBoolean sendExitNoms (const size\_t count, ImqString \* noms [ ] ) ;**

Renvoie des copies des **noms d'exit d'émission** dans les *noms*. Définit tous les *noms* supérieurs à **nombre d'exits d'émission** sur des chaînes nulles. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean setSendExitName(const char \* name = 0) ;**

Définit les **noms d'exit d'émission** sur le *nom* unique. *name* peut être vide ou null. Définit le **nombre d'exits d'émission** sur 1 ou sur zéro. Efface les **données utilisateur d'envoi**. Cette méthode renvoie TRUE en cas de succès

**ImqBoolean setSendExitNames(const size\_t count, const char \* names [ ] ) ;**

Définit les **noms d'exit d'émission** sur *names*. Les valeurs de *noms* individuelles ne doivent pas être vides ou nulles. Définit le **nombre d'exits d'émission** sur *nombre*. Efface les **données utilisateur d'envoi**. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean setSendExitNames(const size\_t count, const ImqString \* names [ ] ) ;**

Définit les **noms d'exit d'émission** sur *names*. Les valeurs de *noms* individuelles ne doivent pas être vides ou nulles. Définit le **nombre d'exits d'émission** sur *nombre*. Efface les **données utilisateur d'envoi**. Cette méthode renvoie TRUE en cas de succès.

**ImqString sendUserData () ;**

Renvoie le premier des **éléments d'envoi de données utilisateur**, le cas échéant. Renvoie une chaîne vide si le **nombre d'exits d'émission** est égal à zéro.

**ImqBoolean sendUserDonnées (const size\_t count, ImqString \* données [ ] ) ;**

Renvoie des copies des éléments **send user data** dans *data*. Définit toutes les *données* supérieures à **nombre d'exits d'émission** sur des chaînes nulles. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean setSendUserData(const char \* data = 0) ;**

Définit **send user data** sur l'élément unique *data*. Si *data* n'est pas null, le **nombre d'exits d'émission** doit être au moins égal à 1. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean setSendUserData(const size\_t count, const char \* data [ ] ) ;**

Définit **send user data** sur *data*. *count* ne doit pas être supérieur au **nombre d'exits d'émission**. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean setSendUserData(const size\_t count, const ImqString \* data [ ] ) ;**

Définit **send user data** sur *data*. *count* ne doit pas être supérieur au **nombre d'exits d'émission**. Cette méthode renvoie TRUE en cas de succès.

**Spécification ImqString sslCipher() const ;**

Renvoie la spécification de chiffrement TLS.

**ImqBoolean setSslCipherSpecification(const char \* nom = 0) ;**

Définit la spécification de chiffrement TLS. Cette méthode renvoie TRUE en cas de succès.

**Authentification MQLONG sslClient() const ;**

Renvoie le type d'authentification de client TLS.

**ImqBoolean setSslClientAuthentication(const MQLONG auth = MQSCA\_REQUIRED) ;**

Définit le type d'authentification du client TLS. Cette méthode renvoie TRUE en cas de succès.

**ImqString sslPeerNom () const ;**

Renvoie le nom de l'homologue TLS.

**ImqBoolean setSslPeerName(const char \* name = 0) ;**

Définit le nom de l'homologue TLS. Cette méthode renvoie TRUE en cas de succès.

**ImqString transactionProgramNom () const ;**

Renvoie le **nom du programme de transaction**.

**ImqBoolean setTransactionProgramName(const char \* name = 0) ;**

Définit le **nom du programme de transaction**. Cette méthode renvoie TRUE en cas de succès.

**MQLONG transportType() const ;**

Renvoie le **type de transport**.

**ImqBoolean setTransportType (const MQLONG type = MQXPT\_LU62 ) ;**

Définit le **type de transport**. Cette méthode renvoie TRUE en cas de succès.

**ImqString userId() const ;**

Renvoie l' **ID utilisateur**.

**ImqBoolean setUserId (const car \* id = 0) ;**

Définit l' **ID utilisateur**. Cette méthode renvoie TRUE en cas de succès.

### Codes raison

- MQRC\_DATA\_LENGTH\_ERROR
- MQRC\_ITEM\_COUNT\_ERREUR
- POINT\_NULL\_MQRC\_NULL
- MQRC\_SOURCE\_BUFFER\_ERROR

## Classe C++ imqCICSBridgeHeader

Cette classe encapsule des fonctions spécifiques de la structure de données MQCIH.

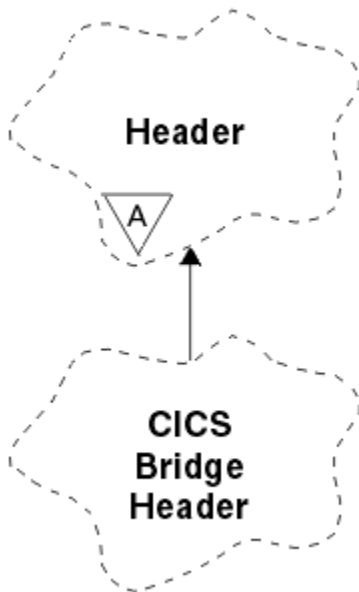


Figure 18. Classe *CICSBridgeHeader* Imq

Les objets de cette classe sont utilisés par les applications qui envoient des messages à CICS bridge via IBM MQ for z/OS.

- [«Attributs d'objet», à la page 1901](#)
- [«Constructeurs», à la page 1904](#)
- [«Méthodes ImqItem surchargées», à la page 1904](#)
- [«Méthodes d'objet \(public\)», à la page 1904](#)
- [«Données d'objet \(protégées\)», à la page 1906](#)
- [«Codes raison», à la page 1907](#)
- [«Codes retour», à la page 1907](#)

### Attributs d'objet

#### Descripteur ADS

Descripteur ADS d'envoi/réception. Cette valeur est définie à l'aide de MQCADSD\_NONE. La valeur initiale est MQCADSD\_NONE. Les valeurs supplémentaires suivantes sont possibles:

- MQCADSD\_NONE
- MQCADSD\_SEND
- MQCADSD\_RECV
- MQCADSD\_MSGFORMAT

#### **caractère de signalisation**

Touche AID. La longueur de la zone doit être MQ\_ATTENTION\_ID\_LENGTH.

#### **authentificateur**

Mot de passe ou mot de passe RACF . La valeur initiale contient des blancs de longueur MQ\_AUTHENTICATOR\_LENGTH.

#### **code de fin anormale de pont**

Code de fin anormale du pont, de longueur MQ\_ABEND\_CODE\_LENGTH. La valeur initiale est de quatre caractères blancs. La valeur renvoyée dans cette zone dépend du code retour. Pour plus d'informations, voir [Tableau 867, à la page 1907](#).

#### **code d'annulation de pont**

Code de transaction de fin anormale de pont. La zone est réservée, doit contenir des blancs et doit avoir une longueur de MQ\_CANCEL\_CODE\_LENGTH.

#### **code achèvement de pont**

Code achèvement, qui peut contenir le code achèvement IBM MQ ou la valeur CICS EIBRESP. La valeur initiale de la zone est MQCC\_OK. La valeur renvoyée dans cette zone dépend du code retour. Pour plus d'informations, voir [Tableau 867, à la page 1907](#).

#### **décalage d'erreur de pont**

Décalage d'erreur de pont. La valeur initiale est zéro. Cet attribut est en lecture seule.

#### **code anomalie de pont**

Code anomalie. Cette zone peut contenir la raison IBM MQ ou la valeur CICS EIBRESP2 . La valeur initiale de la zone est MQRC\_NONE. La valeur renvoyée dans cette zone dépend du code retour. Pour plus d'informations, voir [Tableau 867, à la page 1907](#).

#### **code retour de pont**

Code retour du CICS bridge. La valeur initiale est MQCRC\_OK.

#### **tâche conversationnelle**

Indique si la tâche peut être conversationnelle. La valeur initiale est MQCCT\_NO. Les valeurs supplémentaires suivantes sont possibles:

- MQCCT\_OUI
- MQCCT\_NO

#### **position de curseur**

Position du curseur. La valeur initiale est zéro.

#### **durée de conservation de l'installation**

Heure de libération de la fonction CICS bridge .

#### **fonction telle que**

Attribut de terminal émulé. La longueur de la zone doit être MQ\_FACILITY\_LIKE\_LENGTH.

#### **jeton de fonction**

Valeur du jeton BVT. La longueur de la zone doit être MQ\_FACILITY\_LENGTH. La valeur initiale est MQCFAC\_NONE.

#### **fonction**

Fonction, qui peut contenir le nom d'appel IBM MQ ou la fonction CICS EIBFN. La zone a la valeur initiale MQCFUNC\_NONE, avec la longueur MQ\_FUNCTION\_LENGTH. La valeur renvoyée dans cette zone dépend du code retour. Pour plus d'informations, voir [Tableau 867, à la page 1907](#).

Les valeurs supplémentaires suivantes sont possibles lorsque la **fonction** contient un nom d'appel IBM MQ :

- MQCFUNC\_MQCONN

- MQCFUNC\_MQGET
- MQCFUNC\_MQINQ
- MQCFUNC\_NONE
- MQCFUNC\_MQOPEN
- MQCFUNC\_PUT
- MQCFUNC\_MQPUT1

#### **intervalle d'attente d'obtention**

Intervalle d'attente d'un appel MQGET émis par la tâche CICS bridge . La valeur initiale est MQCGWI\_DEFAULT. La zone s'applique uniquement lorsque **uow control** a la valeur MQCUOWC\_FIRST. Les valeurs supplémentaires suivantes sont possibles:

- MQCGWI\_VALEUR PAR DEFAULT
- MQWI\_ILLIMITÉ

#### **type de lien**

Type de lien. La valeur initiale est MQCLT\_PROGRAM. Les valeurs supplémentaires suivantes sont possibles:

- PROGRAMME MQCL
- TRANSACTION MQCL

#### **identificateur de transaction suivant**

ID de la transaction suivante à joindre. La longueur de la zone doit être MQ\_TRANSACTION\_ID\_LENGTH.

#### **longueur des données de sortie**

Longueur des données COMMAREA. La valeur initiale est MQCODL\_AS\_INPUT.

#### **format de réponse**

Nom de format du message de réponse. La valeur initiale est MQFMT\_NONE avec la longueur MQ\_FORMAT\_LENGTH.

#### **code de démarrage**

Code de début de transaction. La longueur de la zone doit être MQ\_START\_CODE\_LENGTH. La valeur initiale est MQCSC\_NONE. Les valeurs supplémentaires suivantes sont possibles:

- MQCSC\_DEBUT
- Données MQCSC\_STARTDATA
- PUT MQCSC\_TERMINE
- MQCSC\_NONE

#### **statut de fin de tâche**

Statut de fin de la tâche. La valeur initiale est MQCTES\_NOSYNC. Les valeurs supplémentaires suivantes sont possibles:

- MQCTES\_COMMIT
- MQCTES\_BACKOUT
- MQCTES\_ENDTASK
- MQCTES\_NOSYNC

#### **ID transaction**

ID de la transaction à associer. La valeur initiale doit contenir des blancs et doit avoir une longueur de MQ\_TRANSACTION\_ID\_LENGTH. La zone s'applique uniquement lorsque le **contrôle d'unité de travail** a la valeur MQCUOWC\_FIRST ou MQCUOWC\_ONLY.

#### **Contrôle UOW**

Contrôle de l'unité de travail. La valeur initiale est MQCUOWC\_ONLY. Les valeurs supplémentaires suivantes sont possibles:

- MQCUOWC\_FIRST

- MQCUOWC\_MILIEU
- MQCUOWC\_LAST
- MQCUOWC\_ONLY
- MQCUOWC\_COMMIT
- MQCUOWC\_BACKOUT
- MQCUOWC\_CONTINUE

#### version

Numéro de version MQCIH. La valeur initiale est MQCIH\_VERSION\_2. La seule autre valeur prise en charge est MQCIH\_VERSION\_1.

### Constructeurs

#### **ImqCICSBridgeHeader() ;**

Constructeur par défaut.

#### **ImqCICSBridgeHeader(const ImqCICSBridgeHeader & header) ;**

Constructeur de copie.

### Méthodes ImqItem surchargées

#### **virtual ImqBoolean copyOut( ImqMessage & msg ) ;**

Insère une structure de données MQCIH dans la mémoire tampon de message au début, en déplaçant les données de message existantes plus loin et en définissant le format de message sur MQFMT\_CICS.

Pour plus de détails, voir la description de la méthode de classe parent.

#### **virtuel ImqBoolean pasteIn( ImqMessage & msg ) ;**

Lit une structure de données MQCIH à partir de la mémoire tampon de messages. Pour que l'opération aboutisse, le codage de l'objet *msg* doit être MQENC\_NATIVE. Extrayez les messages avec MQGMO\_CONVERT en MQENC\_NATIVE. Pour que l'opération aboutisse, le format ImqMessage doit être MQFMT\_CICS.

Pour plus de détails, voir la description de la méthode de classe parent.

### Méthodes d'objet (public)

#### **void operator = (const ImqCICSBridgeHeader & header) ;**

Copie les données d'instance à partir de l'en-tête, en remplaçant les données d'instance existantes.

#### **MQLONG ADSDescripteur () const ;**

Renvoie une copie du **descripteur ADS**.

#### **void setADSDescriptor(const MQLONG descripteur = MQCADSD\_NONE) ;**

Définit le **descripteur ADS**.

#### **ImqString attentionIdentifieur() const ;**

Renvoie une copie de l' **identificateur d'attention**, complétée avec des blancs de fin jusqu'à la longueur MQ\_ATTENTION\_ID\_LENGTH.

#### **void setAttentionIdentifieur (const char \* data = 0) ;**

Définit l' **identificateur d'attention**, complété par des blancs de fin sur la longueur MQ\_ATTENTION\_ID\_LENGTH. Si aucune *donnée* n'est fournie, réinitialise l' **identificateur d'attention** à la valeur initiale.

#### **ImqString authentificateur () const ;**

Renvoie une copie de l' **authentificateur**, complétée avec des blancs de fin jusqu'à la longueur MQ\_AUTHENTICATOR\_LENGTH.

#### **void setAuthenticator(const char \* data = 0) ;**

Définit l' **authentificateur**, complété par des blancs de fin de longueur MQ\_AUTHENTICATOR\_LENGTH. Si aucune *donnée* n'est fournie, réinitialise l' **authentificateur** à la valeur initiale.



**ImqString bridgeAbendCode () const ;**

Renvoie une copie du **code de fin anormale de pont**, complété par des blancs de fin jusqu'à la longueur MQ\_ABEND\_CODE\_LENGTH.

**ImqString bridgeCancelCode () const ;**

Renvoie une copie du **code d'annulation du pont**, complété par des blancs de fin jusqu'à la longueur MQ\_CANCEL\_CODE\_LENGTH.

**void setBridgeCancelCode(const char \* data = 0) ;**

Définit le **code d'annulation de pont**, complété par des blancs de fin de longueur MQ\_CANCEL\_CODE\_LENGTH. Si aucune *donnée* n'est fournie, réinitialise le **code d'annulation de pont** à la valeur initiale.

**Code bridgeCompletionMQLONG () const ;**

Renvoie une copie du **code achèvement de pont**.

**MQLONG bridgeErrorDécalage () const ;**

Renvoie une copie du **décalage d'erreur de pont**.

**Code bridgeReasonMQLONG () const ;**

Renvoie une copie du **code anomalie du pont**.

**Code bridgeReturnMQLONG () const ;**

Renvoie le **code retour de pont**.

**MQLONG conversationalTask() const ;**

Renvoie une copie de la **tâche conversationnelle**.

**void setConversationalTask (const MQLONG task = MQCCT\_NO) ;**

Définit la **tâche conversationnelle**.

**MQLONG cursorPosition() const ;**

Renvoie une copie de la **position du curseur**.

**void setCursorPosition (const MQLONG position = 0) ;**

Définit la **position du curseur**.

**Durée facilityKeepMQLONG () const ;**

Renvoie une copie de la durée de conservation de la fonction .

**void setFacilityKeepTime(const MQLONG time = 0) ;**

Définit la durée de conservation de la fonction .

**ImqString facilityLike() const ;**

Renvoie une copie de la fonction , **telle que**, complétée avec des blancs de fin jusqu'à la longueur MQ\_FACILITY\_LIKE\_LENGTH.

**void setFacilityLike (const char \* name = 0) ;**

Définit la fonction **comme**, complétée avec des blancs de fin sur la longueur MQ\_FACILITY\_LIKE\_LENGTH. Si aucun *nom* n'est fourni, réinitialise la fonction **comme** la valeur initiale.

**ImqBinary facilityToken() const ;**

Renvoie une copie du **jeton de fonction**.

**ImqBoolean setFacilityJeton (const ImqBinary & jeton ) ;**

Définit le **jeton de fonction**. La **longueur de données** du *jeton* doit être zéro ou MQ\_FACILITY\_LENGTH. Elle renvoie TRUE en cas de succès.

**void setFacilityToken (const MQBYTE8 token = 0) ;**

Définit le **jeton de fonction**. La valeur de *token* peut être zéro, ce qui revient à spécifier MQCFAC\_NONE. Si *token* est différent de zéro, il doit traiter les octets MQ\_FACILITY\_LENGTH des données binaires. Lorsque vous utilisez des valeurs prédéfinies telles que MQCFAC\_NONE, vous devrez peut-être effectuer un transtypage pour garantir une correspondance de signature. Par exemple, (MQBYTE \*) MQCFAC\_NONE.

**Fonction ImqString () const ;**

Renvoie une copie de la **fonction**, complétée avec des blancs de fin jusqu'à la longueur MQ\_FUNCTION\_LENGTH.

**Intervalle getWaitMQLONG () const ;**

Renvoie une copie de l' **intervalle d'attente d'obtention**.

**void setGetWaitInterval(const MQLONG *intervalle* = MQCGWI\_DEFA**

Définit l' **intervalle d'attente d'obtention**.

**MQLONG linkType() const ;**

Renvoie une copie du **type de lien**.

**void setLinkType (const MQLONG *type* = MQCLT\_PROGRAM) ;**

Définit le **type de lien**.

**ImqString nextTransactionIdentificateur () const ;**

Renvoie une copie des données de l' **identificateur de transaction suivant** , complétée avec des blancs de fin jusqu'à la longueur MQ\_TRANSACTION\_ID\_LENGTH.

**MQLONG outputDataLongueur () const ;**

Renvoie une copie de la **longueur des données de sortie**.

**void setOutputDataLength(const MQLONG *longueur* = MQCODL\_AS\_INPUT) ;**

Définit la **longueur des données de sortie**.

**ImqString replyToFormat () const ;**

Renvoie une copie du nom **reply-to format** , complétée avec des blancs de fin jusqu'à la longueur MQ\_FORMAT\_LENGTH.

**void setReplyToFormat(const char \* *name* = 0) ;**

Définit le **format de réponse**, complété avec des blancs de fin sur la longueur MQ\_FORMAT\_LENGTH. Si aucun *nom* n'est fourni, réinitialise le **format de réponse** à la valeur initiale.

**ImqString startCode() const ;**

Renvoie une copie du **code de début**, complétée avec des blancs de fin jusqu'à la longueur MQ\_START\_CODE\_LENGTH.

**void setStartCode (const char \* *data* = 0) ;**

Définit les données du **code de démarrage** , complétées par des blancs de fin sur la longueur MQ\_START\_CODE\_LENGTH. Si aucune *donnée* n'est fournie, réinitialise le **code de début** à la valeur initiale.

**MQLONG taskEndStatut () const ;**

Renvoie une copie du **statut de fin de tâche**.

**ImqString transactionIdentifieur() const ;**

Renvoie une copie des données de l' **identificateur de transaction** , complétée avec des blancs de fin jusqu'à la longueur MQ\_TRANSACTION\_ID\_LENGTH.

**void setTransactionIdentifieur (const char \* *data* = 0) ;**

Définit l' **identificateur de transaction**, rempli avec des blancs de fin sur la longueur MQ\_TRANSACTION\_ID\_LENGTH. Si aucune *donnée* n'est fournie, réinitialise l' **identificateur de transaction** à la valeur initiale.

**MQLONG UOWControl () const ;**

Renvoie une copie du **contrôle d'unité de travail**.

**void setUOWControl(const MQLONG *control* = MQCUOWC\_ONLY) ;**

Définit le **contrôle d'unité de travail**.

**Version MQLONG () const ;**

Renvoie le numéro de **version** .

**ImqBoolean setVersion(const MQLONG *version* = MQCIH\_VERSION\_2) ;**

Définit le numéro de **version** . Elle renvoie TRUE en cas de succès.

## Données d'objet (protégées)

**MQLONG olVersion**

Numéro de version MQCIH maximal pouvant être utilisé dans la mémoire allouée à *opcih*.

### PMQCIH *opcih*

Adresse d'une structure de données MQCIH. La quantité de mémoire allouée est indiquée par *olVersion*.

### Codes raison

- MQRC\_BINARY\_DATA\_LENGTH\_ERROR
- VERSION MQRC\_WRONG\_VERSION

### Codes retour

Tableau 867. Codes retour de la classe *imqCICSBridgeHeader*

Code retour	Fonction	CompCode	Motif	Code de fin anormale
MQCRC_OK				
MQCRC_BRIDGE_ERREUR			MQFB_CICS	
MQCRC_MQ_API_ERREUR	Nom d'appel IBM MQ	IBM MQ CompCode	IBM MQ Motif	
MQCRC_DÉLAI_BRIDGE_ATTENTE	Nom d'appel IBM MQ	IBM MQ CompCode	IBM MQ Motif	
MQCRC_CICS_EXEC_ERROR	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	
MQCRC_SECURITY_ERROR	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	
MQCRC_PROGRAM_NOT_AVAILABLE	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	
MQCRC_TRANSID_NON DISPONIBLE	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	
MQCRC_BRIDGE_ABEND				CICS ABCODE
MQCRC_APPLICATION_ABEND				CICS ABCODE

### Classe C++ *ImqDeadLetterHeader*

Cette classe encapsule les fonctions de la structure de données MQDLH.

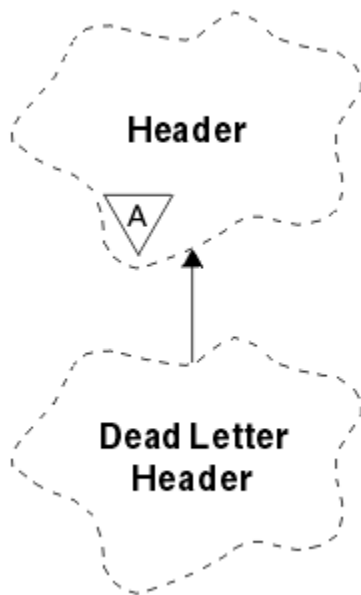


Figure 19. Classe *ImqDeadLetterHeader*

Les objets de cette classe sont généralement utilisés par une application qui rencontre un message qui ne peut pas être traité. Un nouveau message comprenant un en-tête de rebut et le contenu du message est placé dans la file d'attente de rebut et le message est supprimé.

- «Attributs d'objet», à la page 1908
- «Constructeurs», à la page 1909
- «Méthodes *ImqItem* surchargées», à la page 1909
- «Méthodes d'objet (public)», à la page 1909
- «Données d'objet (protégées)», à la page 1910
- «Codes raison», à la page 1910

## Attributs d'objet

### code raison de rebut

Raison pour laquelle le message est arrivé dans la file d'attente des messages non livrés. La valeur initiale est `MQRC_NONE`.

### Nom du gestionnaire de files d'attente de destination

Nom du gestionnaire de files d'attente de destination d'origine. Le nom est une chaîne de longueur `MQ_Q_MGR_NAME_LENGTH`. Sa valeur initiale est `null`.

### nom file d'attente de destination

Nom de la file d'attente de destination d'origine. Le nom est une chaîne de longueur `MQ_Q_NAME_LENGTH`. Sa valeur initiale est `null`.

### Nom d'application d'insertion

Cet attribut indique le nom de l'application ayant inséré le message dans la file d'attente de rebut. Le nom est une chaîne de longueur `MQ_PUT_APPL_NAME_LENGTH`. Sa valeur initiale est `null`.

### Type d'application d'insertion

Type d'application qui a inséré le message dans la file d'attente de rebut. La valeur initiale est zéro.

### Date d'insertion

Date à laquelle le message a été inséré dans la file d'attente de rebut. La date est une chaîne de longueur `MQ_PUT_DATE_LENGTH`. Sa valeur initiale est une chaîne nulle.

### Heure d'insertion

Heure à laquelle le message a été inséré dans la file d'attente de rebut. L'heure est une chaîne de longueur `MQ_PUT_TIME_LENGTH`. Sa valeur initiale est une chaîne nulle.

## Constructeurs

### **ImqDeadLetterHeader();**

Constructeur par défaut.

### **ImqDeadLetterHeader(const ImqDeadLetterHeader & header );**

Constructeur de copie.

## Méthodes ImqItem surchargées

### **virtual ImqBoolean copyOut ( ImqMessage & msg );**

Insère une structure de données MQDLH dans la mémoire tampon du message au début, ce qui déplace les données de message existantes plus loin. Définit le format *msg* sur MQFMT\_DEAD\_LETTER\_HEADER.

Voir la description de la méthode de classe ImqHeader sur la page [«Classe C++ ImqHeader»](#), à la page 1916 pour plus de détails.

### **virtuel ImqBoolean pasteIn ( ImqMessage & msg );**

Lit une structure de données MQDLH à partir de la mémoire tampon de messages.

Pour réussir, le format ImqMessage doit être MQFMT\_DEAD\_LETTER\_HEADER.

Voir la description de la méthode de classe ImqHeader sur la page [«Classe C++ ImqHeader»](#), à la page 1916 pour plus de détails.

## Méthodes d'objet (public)

### **void operator = (const ImqDeadLetterHeader & en-tête );**

Copie les données d'instance à partir de l' *en-tête*, en remplaçant les données d'instance existantes.

### **MQLONG deadLetterReasonCode () const ;**

Renvoie le code raison de rebut.

### **void setDeadLetterReasonCode (const MQLONG reason );**

Définit le code raison de la lettre morte.

### **ImqString destinationQueueManagerName () const ;**

Renvoie le nom du gestionnaire de files d'attente de destination, sans les blancs de fin.

### **void setDestinationQueueManagerNom (const char \* name );**

Définit le nom du gestionnaire de files d'attente de destination. Tronque les données dont la longueur est supérieure à MQ\_Q\_MGR\_NAME\_LENGTH (48 caractères).

### **ImqString destinationQueueNom () const ;**

Renvoie une copie du nom de la file d'attente de destination, sans les blancs de fin.

### **void setDestinationQueueName (const char \* name );**

Définit le nom de la file d'attente de destination. Tronque les données dont la longueur est supérieure à MQ\_Q\_NAME\_LENGTH (48 caractères).

### **ImqString putApplicationNom () const ;**

Renvoie une copie du nom de l'application d'insertion, sans les blancs de fin.

### **void setPutApplicationName (const char \* name = 0 );**

Définit le nom de l'application d'insertion. Tronque les données dont la longueur est supérieure à MQ\_PUT\_APPL\_NAME\_LENGTH (28 caractères).

### **MQLONG putApplicationType () const ;**

Renvoie le type d'application d'insertion.

### **void setPutApplicationType (const MQLONG type = MQAT\_NO\_CONTEXT );**

Définit le type d'application d'insertion.

### **ImqString putDate () const ;**

Renvoie une copie de la date d'insertion, sans les blancs de fin.

**void setPutDate (const char \* date = 0) ;**

Définit la date d'insertion. Tronque les données dont la longueur est supérieure à MQ\_PUT\_DATE\_LENGTH (8 caractères).

**ImqString putTime () const ;**

Renvoie une copie de l'heure d'insertion, débarrassée des blancs de fin.

**void setPutTime (const char \* time = 0) ;**

Définit l'heure d'insertion. Tronque les données dont la longueur est supérieure à MQ\_PUT\_TIME\_LENGTH (8 caractères).

## Données d'objet (protégées)

**MQDLH omqdlh**

Structure de données MQDLH.

## Codes raison

- MQRC\_INCONSISTENT\_FORMAT
- MQRC\_STRUC\_ID\_ERROR
- ERREUR\_CODAGE\_MQRC\_ERREUR

## ImqDistributionRépertorie la classe C++

Cette classe encapsule une liste de distribution dynamique qui fait référence à une ou plusieurs files d'attente dans le but d'envoyer un ou plusieurs messages à plusieurs destinations.

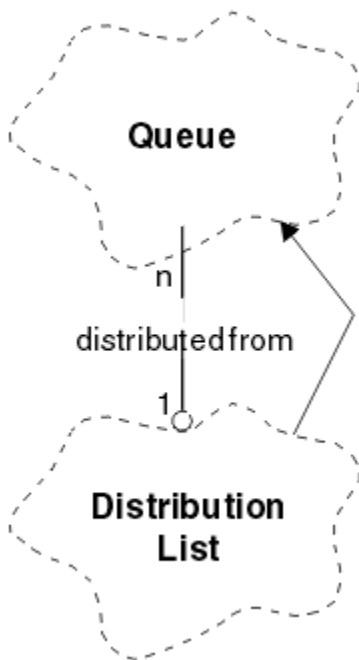


Figure 20. Classe de liste ImqDistribution

- [«Attributs d'objet», à la page 1911](#)
- [«Constructeurs», à la page 1911](#)
- [«Méthodes d'objet \(public\)», à la page 1911](#)
- [«Méthodes d'objet \(protégées\)», à la page 1911](#)

## Attributs d'objet

### première file d'attente répartie

Premier d'un ou de plusieurs objets de classe, sans ordre particulier, dans lequel la **référence de liste de distribution** traite cet objet.

Au départ, il n'y a pas d'objets de ce type. Pour que l'ouverture d'une liste `ImqDistribution` aboutisse, il doit y avoir au moins un objet de ce type.

**Remarque :** Lorsqu'un objet de liste `ImqDistribution` est ouvert, tous les objets ouverts qui y font référence sont automatiquement fermés.

## Constructeurs

### `ImqDistributionList () ;`

Constructeur par défaut.

### `ImqDistributionList ( const ImqDistributionList & list ) ;`

Constructeur de copie.

## Méthodes d'objet (public)

### `void operator = ( const ImqDistributionList & list ) ;`

Tous les objets qui font référence à **cet objet** sont déréférencés avant la copie. Aucun objet ne fera référence à **cet** objet après l'appel de cette méthode.

### `* firstDistributedQueue () const ;`

Renvoie la **première file d'attente répartie**.

## Méthodes d'objet (protégées)

### `void setFirstDistributedQueue ( * file d'attente = 0 ) ;`

Définit la **première file d'attente répartie**.

## Classe C++ `ImqError`

Cette classe abstraite fournit des informations sur les erreurs associées à un objet.

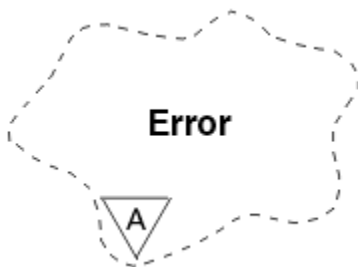


Figure 21. Classe `ImqError`

- [«Attributs d'objet»](#), à la page 1911
- [«Constructeurs»](#), à la page 1912
- [«Méthodes d'objet \(public\)»](#), à la page 1912
- [«Méthodes d'objet \(protégées\)»](#), à la page 1912
- [«Codes raison»](#), à la page 1912

## Attributs d'objet

### code achèvement

Code achèvement le plus récent. La valeur initiale est zéro. Les valeurs supplémentaires suivantes sont possibles:

- MQCC\_OK
- MQCC\_WARNING
- MQCC\_FAILED

#### **code anomalie**

Code anomalie le plus récent. La valeur initiale est zéro.

### **Constructeurs**

#### **ImqError();**

Constructeur par défaut.

#### **ImqError( const ImqError & erreur );**

Constructeur de copie.

### **Méthodes d'objet (public)**

#### **void operator = ( const ImqError & error );**

Copie les données d'instance à partir de *error*, en remplaçant les données d'instance existantes.

#### **void clearErrorCodes ();**

Définit le **code achèvement** et le **code anomalie** sur zéro.

#### **MQLONG completionCode () const ;**

Renvoie le **code achèvement**.

#### **MQLONG reasonCode () const ;**

Renvoie le **code anomalie**.

### **Méthodes d'objet (protégées)**

#### **ImqBoolean checkReadPointeur ( const void \* *pointeur*, const size\_t *longueur* );**

Vérifie que la combinaison du pointeur et de la longueur est valide pour l'accès en lecture seule et renvoie TRUE si l'opération aboutit.

#### **ImqBoolean checkWritePointer ( const void \* *pointer*, const size\_t *longueur* );**

Vérifie que la combinaison du pointeur et de la longueur est valide pour l'accès en lecture-écriture et renvoie TRUE en cas de réussite.

#### **void setCompletionCode ( const MQLONG *code* = 0 );**

Définit le **code achèvement**.

#### **void setReasonCode ( const MQLONG *code* = 0 );**

Définit le **code anomalie**.

### **Codes raison**

- MQRC\_BUFFER\_ERROR

## **Classe C++ ImqGetMessageOptions**

Cette classe encapsule la structure de données MQGMO



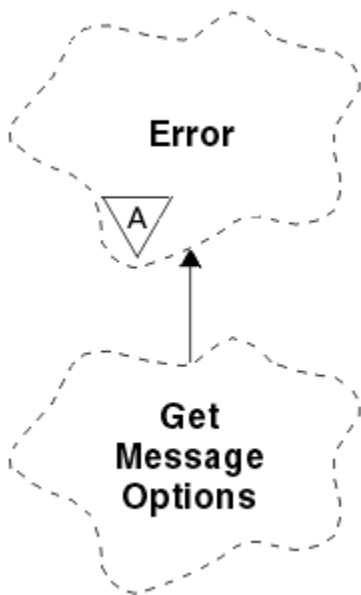


Figure 22. Classe *ImqGetMessageOptions*

- «Attributs d'objet», à la page 1913
- «Constructeurs», à la page 1914
- «Méthodes d'objet (public)», à la page 1915
- «Méthodes d'objet (protégées)», à la page 1916
- «Données d'objet (protégées)», à la page 1916
- «Codes raison», à la page 1916

## Attributs d'objet

### statut de groupe

Statut d'un message pour un groupe de messages. La valeur initiale est MQGS\_NOT\_IN\_GROUP. Les valeurs supplémentaires suivantes sont possibles:

- MQGS\_MSG\_IN\_GROUPE
- MQGS\_LAST\_MSG\_IN\_GROUP

### options de mise en correspondance

Options de sélection des messages entrants. La valeur initiale est MQMO\_MATCH\_MSG\_ID | MQMO\_MATCH\_CORREL\_ID. Les valeurs supplémentaires suivantes sont possibles:

- ID\_GROUPE\_MQMO
- NUMERO MQMO\_MATCH\_MSG\_SEQ\_NO
- MQMO\_MATCH\_OFFSET
- JETON\_MSG\_MQMO\_TOKEN
- MQMO\_AUCUN

### jeton de message

Jeton de message. Une valeur binaire (MQBYTE16) de longueur MQ\_MSG\_TOKEN\_LENGTH. La valeur initiale est MQMTOK\_NONE.

### options

Options applicables à un message. La valeur initiale est MQGMO\_NO\_WAIT. Les valeurs supplémentaires suivantes sont possibles:

- MQGMO\_WAIT
- MQGMO\_SYNCPOINT

- MQGMO\_SYNCPOINT\_IF\_PERSISTENT
- MQGMO\_NO\_SYNCPOINT
- MQGMO\_MARK\_SKIP\_BACKOUT
- MQGMO\_BROWSE\_FIRST
- MQGMO\_BROWSE\_NEXT
- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
- MQGMO\_MSG\_UNDER\_CURSOR
- MQGMO\_LOCK
- MQGMO\_UNLOCK
- MQGMO\_ACCEPT\_TRUNCATED\_MSG
- MQGMO\_SET\_SIGNAL
- MQGMO\_FAIL\_IF QUIESCING
- MQGMO\_CONVERT
- MQGMO\_LOGICAL\_ORDER
- MQGMO\_COMPLETE\_MSG
- MQGMO\_ALL\_MSGS\_AVAILABLE
- MQGMO\_ALL\_SEGMENTS\_AVAILABLE
- MQGMO\_NONE

#### **Nom de file d'attente résolu**

Nom de file d'attente résolu. Cet attribut est en lecture seule. Les noms ne comportent jamais plus de 48 caractères et peuvent être complétés par des valeurs nulles. La valeur initiale est une chaîne nulle.

#### **longueur renvoyée**

Longueur renvoyée. La valeur initiale est MQRL\_UNDEFINED. Cet attribut est en lecture seule.

#### **segmentation**

Possibilité de segmenter un message. La valeur initiale est MQSEG\_INHIBÉE. La valeur supplémentaire, MQSEG\_ALLOWED, est possible.

#### **statut de segment**

Statut de segmentation d'un message. La valeur initiale est MQSS\_NOT\_A\_SEGMENT. Les valeurs supplémentaires suivantes sont possibles:

- SEGMENT\_MQSS\_SEGMENT
- SEGMENT\_LAST\_MQSS\_

#### **participation au point de synchronisation**

TRUE lorsque les messages sont extraits sous le contrôle de point de synchronisation.

#### **intervalle d'attente**

Durée pendant laquelle la méthode get de la classe s'interrompt en attendant l'arrivée d'un message approprié, s'il n'en existe pas déjà un. La valeur initiale est zéro, ce qui entraîne une attente indéfinie. La valeur supplémentaire, MQWI\_UNLIMITED, est possible. Cet attribut est ignoré sauf si les options incluent MQGMO\_WAIT.

## **Constructeurs**

### **ImqGetMessageOptions( );**

Constructeur par défaut.

### **ImqGetMessageOptions(const ImqGetMessageOptions & gmo );**

Constructeur de copie.

## Méthodes d'objet (public)

**void operator = (const ImqGetMessageOptions & gmo ) ;**

Copie les données d'instance à partir de *gmo*, en remplaçant les données d'instance existantes.

**MQCHAR groupStatus () const ;**

Renvoie le statut du groupe.

**void setGroupStatus (const MQCHAR status ) ;**

Définit le statut du groupe.

**MQLONG matchOptions () const ;**

Renvoie les options de correspondance.

**void setMatchOptions (const MQLONG options ) ;**

Définit les options de correspondance.

**ImqBinary messageToken() const ;**

Renvoie le jeton de message.

**ImqBoolean setMessageToken (const ImqBinary & token ) ;**

Définit le jeton de message. La longueur des données du *jeton* doit être égale à zéro ou MQ\_MSG\_TOKEN\_LENGTH. Cette méthode renvoie TRUE en cas de succès.

**void setMessageToken (const MQBYTE16 token = 0) ;**

Définit le jeton de message. La valeur de *token* peut être égale à zéro, ce qui revient à spécifier MQMTOK\_NONE. Si *token* est différent de zéro, il doit traiter les octets MQ\_MSG\_TOKEN\_LENGTH des données binaires.

Lorsque vous utilisez des valeurs prédéfinies, telles que MQMTOK\_NONE, vous n'avez peut-être pas besoin d'effectuer un transtypage pour garantir une correspondance de signature, par exemple (MQBYTE \*) MQMTOK\_NONE.

**Options MQLONG () const ;**

Renvoie les options.

**void setOptions (const MQLONG options ) ;**

Définit les options, y compris la valeur de participation du point de synchronisation.

**ImqString resolvedQueueNom () const ;**

Renvoie une copie du nom de la file d'attente résolue.

**MQLONG returnedLength() const ;**

Renvoie la longueur renvoyée.

**Segmentation MQCHAR () const ;**

Renvoie la segmentation.

**void setSegmentation (const MQCHAR valeur ) ;**

Définit la segmentation.

**MQCHAR segmentStatus () const ;**

Renvoie le statut du segment.

**void setSegmentStatus (const MQCHAR status ) ;**

Définit le statut du segment.

**ImqBoolean syncPointParticipation () const ;**

Renvoie la valeur de participation du point de synchronisation, qui est TRUE si les options incluent MQGMO\_SYNCPOINT ou MQGMO\_SYNCPOINT\_IF\_PERSISTENT.

**void setSyncPointParticipation (const ImqBoolean sync ) ;**

Définit la valeur de participation du point de synchronisation. Si *sync* est TRUE, modifie les options pour inclure MQGMO\_SYNCPOINT et exclure à la fois MQGMO\_NO\_SYNCPOINT et MQGMO\_SYNCPOINT\_IF\_PERSISTENT. Si *sync* a pour valeur FALSE, modifie les options d'inclusion de MQGMO\_NO\_SYNCPOINT et d'exclusion de MQGMO\_SYNCPOINT et de MQGMO\_SYNCPOINT\_IF\_PERSISTENT.

**MQLONG waitInterval () const ;**

Renvoie l'intervalle d'attente.

**void setWaitInterval (const MQLONG *intervalle* ) ;**

Définit l'intervalle d'attente.

### Méthodes d'objet (protégées)

**static void setVersionSupported (const MQLONG) ;**

Définit la version MQGMO. La valeur par défaut est MQGMO\_VERSION\_3.

### Données d'objet (protégées)

#### **MQGMO *omqgmo***

Structure de données MQGMO version 2. Accès aux zones MQGMO prises en charge pour MQGMO\_VERSION\_2 uniquement.

#### **PMQGMO *opgmo***

Adresse d'une structure de données MQGMO. Le numéro de version de cette adresse est indiqué dans *olVersion*. Vérifiez le numéro de version avant d'accéder aux zones MQGMO pour vous assurer qu'elles sont présentes.

#### **MQLONG *olVersion***

Numéro de version de la structure de données MQGMO traitée par *opgmo*.

### Codes raison

- MQRC\_BINARY\_DATA\_LENGTH\_ERROR

## Classe C++ ImqHeader

Cette classe abstraite encapsule les fonctions communes de la structure de données MQDLH.

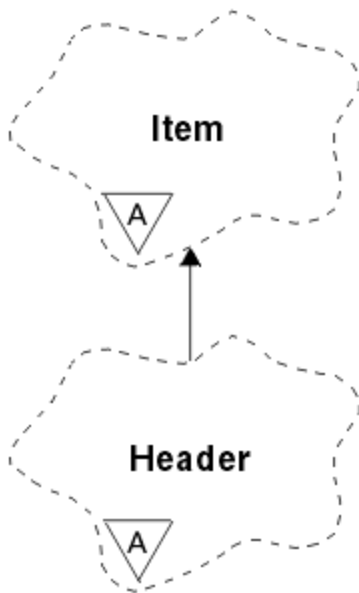


Figure 23. Classe *ImqHeader*

- «Attributs d'objet», à la page [1916](#)
- «Constructeurs», à la page [1917](#)
- «Méthodes d'objet (public)», à la page [1917](#)

### Attributs d'objet

#### **jeu de caractères**

Identificateur du jeu de caractères codés d'origine. Initialement MQCCSI\_Q\_MGR.

**codage**

Codage d'origine. Initialement MQENC\_NATIVE.

**format**

Format d'origine. Initialement MQFMT\_NONE.

**indicateurs d'en-tête**

Les valeurs initiales sont les suivantes:

- Zéro pour les objets de la classe ImqDeadLetterHeader
- MQIIH\_NONE pour les objets de la classe ImqIMSBridgeHeader
- MQRMHF\_LAST pour les objets de la classe d'en-tête ImqReference
- MQCIH\_NONE pour les objets de la classe ImqCICSBridgeHeader
- MQWIH\_NONE pour les objets de la classe d'en-tête ImqWork

**Constructeurs****ImqHeader();**

Constructeur par défaut.

**ImqHeader( const ImqHeader & header );**

Constructeur de copie.

**Méthodes d'objet (public)****void operator = ( const ImqHeader & header );**

Copie les données d'instance à partir de l' *en-tête*, en remplaçant les données d'instance existantes.

**virtual MQLONG characterSet () const ;**

Renvoie le **jeu de caractères**.

**virtual void setCharacterSet ( const MQLONG ccsid = MQCCSI\_Q\_MGR );**

Définit le **jeu de caractères**.

**codage () virtuel MQLONG const ;**

Renvoie le **codage**.

**virtual void setEncoding ( const MQLONG encoding = MQENC\_NATIVE );**

Définit le **codage**.

**virtual ImqString format () const ;**

Renvoie une copie du **format**, y compris les blancs de fin.

**virtual void setFormat ( const char \* name = 0 );**

Définit le **format**, complété par 8 caractères avec des blancs de fin.

**virtual MQLONG headerFlags () const ;**

Renvoie les **indicateurs d'en-tête**.

**virtual void setHeaderFlags ( const MQLONG indicateurs = 0 );**

Définit les **indicateurs d'en-tête**.

**Classe C++ imqIMSBridgeHeader**

Cette classe encapsule les fonctions de la structure de données MQIIH.

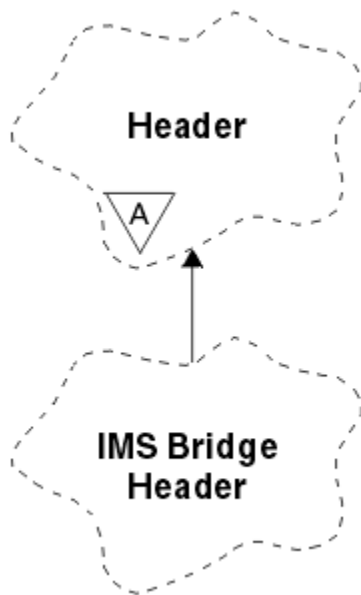


Figure 24. Classe *IMSBridgeHeader* Imq

Les objets de cette classe sont utilisés par les applications qui envoient des messages à la passerelle IMS via IBM MQ for z/OS.

**Remarque :** Le jeu de caractères et le codage *ImqHeader* doivent avoir des valeurs par défaut et ne doivent pas être définis sur d'autres valeurs.

- «Attributs d'objet», à la page 1918
- «Constructeurs», à la page 1919
- «Méthodes *ImqItem* surchargées», à la page 1919
- «Méthodes d'objet (public)», à la page 1919
- «Données d'objet (protégées)», à la page 1920
- «Codes raison», à la page 1920

## Attributs d'objet

### authentificateur

Mot de passe ou mot de passe RACF , de longueur `MQ_AUTHENTICATOR_LENGTH`. La valeur initiale est `MQIAUT_NONE`.

### Mode de validation

Mode de validation. Pour plus d'informations sur les modes de validation IMS , voir le *Guide d'utilisation d'OTMA* . La valeur initiale est `MQICM_COMMIT_THEN_SEND`. La valeur supplémentaire, `MQICM_SEND_THEN_COMMIT`, est possible.

### Substitution de terminal logique

Remplacement de terminal logique, de longueur `MQ_LTERM_OVERRIDE_LENGTH`. La valeur initiale est une chaîne nulle.

### Nom de mappe des services de structuration de messages

Nom de mappe MFS, de longueur `MQ_MFS_MAP_NAME_LENGTH`. La valeur initiale est une chaîne nulle.

### format de réponse

Format de toute réponse, de longueur `MQ_FORMAT_LENGTH`. La valeur initiale est `MQFMT_NONE`.

### portée de sécurité

Portée du traitement de la sécurité IMS . La valeur initiale est `MQISS_CHECK`. La valeur supplémentaire, `MQISS_FULL`, est possible.

## **ID instance de transaction**

Identité de l'instance de transaction, valeur binaire (MQBYTE16) de longueur MQ\_TRAN\_INSTANCE\_ID\_LENGTH. La valeur initiale est MQITII\_NONE.

## **ETAT DE TRANSACTION**

Etat de la conversation IMS . La valeur initiale est MQITS\_NOT\_IN\_CONVERSATION. La valeur supplémentaire, MQITS\_IN\_CONVERSATION, est possible.

## **Constructeurs**

### **ImqIMSBridgeHeader() ;**

Constructeur par défaut.

### **ImqIMSBridgeHeader(const ImqIMSBridgeHeader & header) ;**

Constructeur de copie.

## **Méthodes ImqItem surchargées**

### **virtual ImqBoolean copyOut ( ImqMessage & msg ) ;**

Insère une structure de données MQIIH dans la mémoire tampon de message au début, déplaçant les données de message existantes plus loin. Définit le format *msg* sur MQFMT\_IMS.

Pour plus de détails, voir la description de la méthode de la classe parent.

### **virtuel ImqBoolean pasteIn ( ImqMessage & msg ) ;**

Lit une structure de données MQIIH à partir de la mémoire tampon de messages.

Pour que l'opération aboutisse, le codage de l'objet *msg* doit être MQENC\_NATIVE. Extrayez les messages avec MQGMO\_CONVERT en MQENC\_NATIVE.

Pour que l'opération aboutisse, le format ImqMessage doit être MQFMT\_IMS.

Pour plus de détails, voir la description de la méthode de la classe parent.

## **Méthodes d'objet (public)**

### **void operator = (const ImqIMSBridgeHeader & header) ;**

Copie les données d'instance à partir de l' *en-tête*, en remplaçant les données d'instance existantes.

### **ImqString authentificateur () const ;**

Renvoie une copie de l'authentificateur, complétée avec des blancs de fin jusqu'à la longueur MQ\_AUTHENTICATOR\_LENGTH.

### **void setAuthenticator (const char \* name) ;**

Définit l'authentificateur.

### **MQCHAR commitMode () const ;**

Renvoie le mode de validation.

### **void setCommitMode (const MQCHAR mode) ;**

Définit le mode de validation.

### **ImqString logicalTerminalRemplacer () const ;**

Renvoie une copie de la substitution de terminal logique.

### **void setLogicalTerminalOverride (const char \* override) ;**

Définit le remplacement du terminal logique.

### **ImqString messageFormatServicesMapNom () const ;**

Renvoie une copie du nom de mappe des services de format de message.

### **void setMessageFormatServicesMapName (const char \* name) ;**

Définit le nom de mappe des services de format de message.

### **ImqString replyToFormat () const ;**

Renvoie une copie du format de réponse, complétée avec des blancs de fin de longueur MQ\_FORMAT\_LENGTH.

**void setReplyToFormat (const char \* format ) ;**

Définit le format de réponse, complété par des blancs de fin sur la longueur MQ\_FORMAT\_LENGTH.

**MQCHAR securityScope () const ;**

Renvoie la portée de sécurité.

**void setSecurityScope (const MQCHAR scope ) ;**

Définit la portée de sécurité.

**ID ImqBinary transactionInstance() const ;**

Renvoie une copie de l'ID d'instance de transaction.

**ImqBoolean setTransactionInstanceId (const ImqBinary & id ) ;**

Définit l'ID d'instance de transaction. La longueur des données du *jeton* doit être égale à zéro ou MQ\_TRAN\_INSTANCE\_ID\_LENGTH. Cette méthode renvoie TRUE en cas de succès.

**void setTransactionInstanceId (const MQBYTE16 id = 0) ;**

Définit l'ID d'instance de transaction. *id* peut être égal à zéro, ce qui revient à spécifier MQITII\_NONE. Si *id* est différent de zéro, il doit traiter les octets MQ\_TRAN\_INSTANCE\_ID\_LENGTH des données binaires. Lorsque vous utilisez des valeurs prédéfinies telles que MQITII\_NONE, vous pouvez être amené à effectuer un transtypage pour garantir une correspondance de signature, par exemple (MQBYTE \*) MQITII\_NONE.

**MQCHAR transactionState () const ;**

Renvoie l'état de la transaction.

**void setTransactionState (const MQCHAR state ) ;**

Définit l'état de la transaction.

## Données d'objet (protégées)

**MQIIH omqiih**

Structure de données MQIIH.

## Codes raison

- MQRC\_BINARY\_DATA\_LENGTH\_ERROR
- MQRC\_INCONSISTENT\_FORMAT
- ERREUR\_CODAGE\_MQRC\_ERREUR
- MQRC\_STRUC\_ID\_ERROR

## Classe C++ ImqItem

Cette classe abstraite représente un élément, peut-être un parmi plusieurs, dans un message.



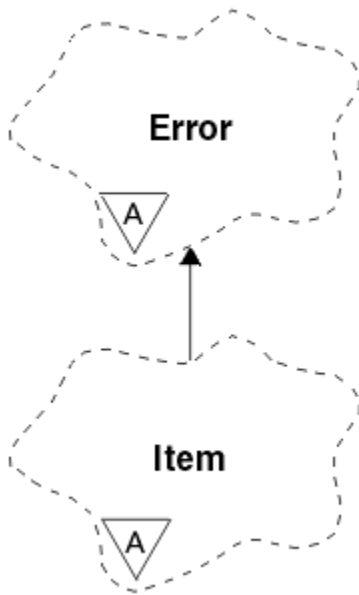


Figure 25. Classe *ImqItem*

Les éléments sont concaténés dans une mémoire tampon de messages. Chaque spécialisation est associée à une structure de données particulière qui commence par un ID de structure.

Les méthodes polymorphes de cette classe abstraite permettent de copier des éléments vers et depuis des messages. La classe *ImqMessage* **readItem** et les méthodes **writeItem** fournissent un autre style d'appel de ces méthodes polymorphes plus naturel pour les programmes d'application.

- «Attributs d'objet», à la page 1921
- «Constructeurs», à la page 1921
- «Méthodes de classe (public)», à la page 1921
- «Méthodes d'objet (public)», à la page 1922
- «Codes raison», à la page 1922

## Attributs d'objet

### ID structure

Chaîne de quatre caractères au début de la structure de données. Cet attribut est en lecture seule. Tenez compte de cet attribut pour les classes dérivées. Il n'est pas inclus automatiquement.

## Constructeurs

### **ImqItem();**

Constructeur par défaut.

### **ImqItem( const ImqItem & item );**

Constructeur de copie.

## Méthodes de classe (public)

### **static ImqBoolean structureIds ( const char \* structure-id-to-test, const ImqMessage & msg );**

Renvoie TRUE si l' **ID de structure** de l'élément *ImqItem* suivant dans le message *msg* entrant est identique à *structure-id-to-test*. L'élément suivant est identifié comme partie de la mémoire tampon de messages actuellement traitée par le **pointeur de données** *ImqCache*. Cette méthode s'appuie sur l' **ID de structure** et n'est donc pas garantie de fonctionner pour toutes les classes dérivées *ImqItem*.

## Méthodes d'objet (public)

**void operator = ( const ImqItem & item ) ;**

Copie les données d'instance à partir de *item*, en remplaçant les données d'instance existantes.

**virtual ImqBoolean copyOut ( ImqMessage & msg ) = 0 ;**

Écrit cet objet en tant qu'élément suivant dans une mémoire tampon de messages sortants, en l'ajoutant à tous les éléments existants. Si l'opération d'écriture aboutit, augmente la **longueur de données** ImqCache . Cette méthode renvoie TRUE en cas de succès.

Remplacez cette méthode pour utiliser une sous-classe spécifique.

**virtuel ImqBoolean pasteIn ( ImqMessage & msg ) = 0 ;**

Lit cet objet *de façon destructive* à partir de la mémoire tampon des messages entrants. La lecture est destructive dans la mesure où le **pointeur de données** ImqCache est déplacé. Toutefois, le contenu de la mémoire tampon reste le même, de sorte que les données peuvent être relectées en réinitialisant le **pointeur de données** ImqCache .

La (sous-) classe de cet objet doit être cohérente avec l' **ID de structure** qui se trouve ensuite dans la mémoire tampon de messages de l'objet *msg* .

Le **codage** de l'objet *msg* doit être MQENC\_NATIVE. Il est recommandé d'extraire les messages avec l'option ImqMessage **encoding** définie sur MQENC\_NATIVE et avec les options ImqGetMessageOptions incluant MQGMO\_CONVERT.

Si l'opération de lecture aboutit, la **longueur de données** ImqCache est réduite. Cette méthode renvoie TRUE en cas de succès.

Remplacez cette méthode pour utiliser une sous-classe spécifique.

## Codes raison

- ERREUR\_CODAGE\_MQRC\_ERREUR
- MQRC\_STRUC\_ID\_ERROR
- MQRC\_INCONSISTENT\_FORMAT
- MQRC\_INSUFFISANT\_BUFFER
- MQRC\_DONNÉES\_INSUFFISANTES

## Classe C++ ImqMessage

Cette classe encapsule une structure de données MQMD et gère également la construction et la reconstruction des données de message.

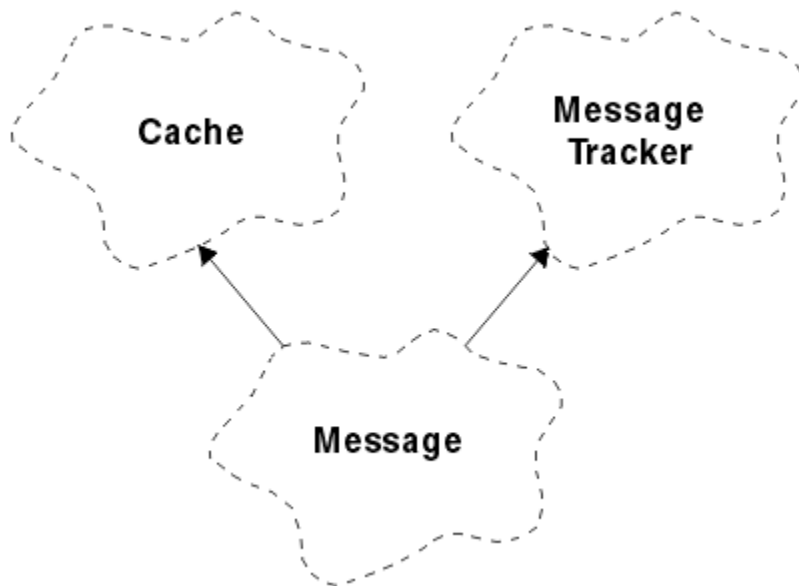


Figure 26. Classe *ImqMessage*

- «Attributs d'objet», à la page [1923](#)
- «Constructeurs», à la page [1927](#)
- «Méthodes d'objet (public)», à la page [1927](#)
- «Méthodes d'objet (protégées)», à la page [1929](#)
- «Données d'objet (protégées)», à la page [1929](#)

## Attributs d'objet

### Données d'ID application

Informations d'identité associées à un message. La valeur initiale est une chaîne nulle.

### Données d'origine de l'application

Informations d'origine associées à un message. La valeur initiale est une chaîne nulle.

### nombre d'annulations

Nombre de fois où un message a été provisoirement extrait, puis annulé. La valeur initiale est zéro. Cet attribut est en lecture seule.

### jeu de caractères

ID de jeu de caractères codés. La valeur initiale est MQCCSI\_Q\_MGR. Les valeurs supplémentaires suivantes sont possibles:

- MQCCSI\_HÉRITER
- MQCCSI\_IMBRIQUÉ

Vous pouvez également utiliser un ID de jeu de caractères codés de votre choix. Pour plus d'informations à ce sujet, voir «Conversion de page de code», à la page [984](#).

### codage

Codage machine des données de message. La valeur initiale est MQENC\_NATIVE.

### expiration

Quantité avec contrainte horaire qui contrôle la durée pendant laquelle IBM MQ conserve un message non extrait avant de le supprimer. La valeur initiale est MQEI\_UNLIMITED.

### format

Nom du format (modèle) qui décrit la présentation des données dans la mémoire tampon. Les noms de plus de huit caractères sont tronqués à huit caractères. Les noms sont toujours complétés avec des blancs à huit caractères. La valeur de la constante initiale est MQFMT\_NONE. Les constantes supplémentaires suivantes sont possibles:

- MQFMT\_ADMIN
- MQFMT\_CICS
- MQFMT\_COMMAND\_1
- MQFMT\_COMMAND\_2
- MQFMT\_DEAD\_LETTER\_HEADER
- MQFMT\_DIST\_HEADER
- MQFMT\_EVENT
- MQFMT\_IMS
- MQFMT\_IMS\_VAR\_STRING
- MQFMT\_MD\_EXTENSION
- MQFMT\_PCF
- MQFMT\_REF\_MSG\_HEADER
- MQFMT\_RF\_HEADER
- MQFMT\_STRING
- MQFMT\_TRIGGER
- MQFMT\_WORK\_INFO\_HEADER
- MQFMT\_XMIT\_Q\_HEADER

Vous pouvez également utiliser une chaîne spécifique à l'application de votre choix. Pour plus d'informations à ce sujet, voir la zone [«Format \(MQCHAR8\) pour MQMD»](#), à la page 465 du descripteur de message (MQMD).

#### **Indicateurs de message**

Informations de contrôle de segmentation. La valeur initiale est MQMF\_SEGMENTATION\_INHIBÉE. Les valeurs supplémentaires suivantes sont possibles:

- MQMF\_SEGMENTATION\_ALLOWED
- GROUPE MQMF\_MSG\_IN\_GROUP
- MQMF\_LAST\_MSG\_IN\_GROUP
- SEGMENT\_MQMF
- SEGMENT\_DERNIER\_MQMF\_SEGMENT
- MQMF\_AUCUN

#### **type de message**

Catégorisation générale d'un message. La valeur initiale est MQMT\_DATAGRAM. Les valeurs supplémentaires suivantes sont possibles:

- MQMT\_SYSTEM\_FIRST
- MQMT\_SYSTEM\_LAST
- MQMT\_DATAGRAM
- MQMT\_REQUEST
- MQMT\_REPLY
- MQMT\_REPORT
- MQMT\_APPL\_FIRST
- MQMT\_APPL\_LAST

Vous pouvez également utiliser une valeur spécifique à l'application de votre choix. Pour plus d'informations à ce sujet, voir la zone [«MsgType \(MQLONG\) pour MQMD»](#), à la page 454 du descripteur de message (MQMD).

#### **décalage**

Informations de décalage. La valeur initiale est zéro.

### Longueur d'origine

Longueur d'origine d'un message segmenté. La valeur initiale est MQOL\_UNDEFINED.

### persistance

Indique que le message est important et qu'il doit être sauvegardé à tout moment à l'aide du stockage de persistance. Cette option implique une baisse des performances. La valeur initiale est MQPER\_PERSISTENCE\_AS\_Q\_DEF. Les valeurs supplémentaires suivantes sont possibles:

- MQPER\_PERSISTANT
- MQPER\_NON\_PERSISTENT

### priority

Priorité relative pour la transmission et la distribution. Les messages de même priorité sont généralement distribués dans la même séquence qu'ils ont été fournis (bien qu'il existe plusieurs critères qui doivent être remplis pour garantir cela). La valeur initiale est MQPRI\_PRIORITY\_AS\_Q\_DEF.

### Validation de propriété

Indique si la validation des propriétés doit avoir lieu lorsqu'une propriété du message est définie. La valeur initiale est MQCMHO\_DEFAULT\_VALIDATION. Les valeurs supplémentaires suivantes sont possibles:

- MQCMHO\_VALIDATION
- MQCMHO\_NO\_VALIDATION

Les méthodes suivantes agissent sur la **validation de propriété**:

**MQLONG propertyValidation() const ;**

Renvoie l'option **property validation** .

**void setPropertyValidation (const MQLONG option ) ;**

Définit l'option **property validation** .

### Nom d'application d'insertion

Nom de l'application qui a inséré un message. La valeur initiale est une chaîne nulle.

### Type d'application d'insertion

Type d'application qui a inséré un message. La valeur initiale est MQAT\_NO\_CONTEXT. Les valeurs supplémentaires suivantes sont possibles:

- MQAT\_AIX
- MQAT\_CICS
- MQAT\_CICS\_BRIDGE
- MQAT\_DOS
- MQAT\_IMS
- MQAT\_IMS\_BRIDGE
- MQAT\_MVS
- MQAT\_NOTES\_AGENT
- MQAT\_OS2
- MQAT\_OS390
- MQAT\_OS400
- MQAT\_QMGR
- MQAT\_UNIX
- MQAT\_WINDOWS
- MQAT\_WINDOWS\_NT
- MQAT\_XCF
- MQAT\_PAR DEFALT

- MQAT\_INCONNU
- MQAT\_USER\_FIRST
- MQAT\_USER\_LAST

Vous pouvez également utiliser une chaîne spécifique à l'application de votre choix. Pour plus d'informations à ce sujet, voir la zone «PutApplType (MQLONG) pour MQMD», à la page 481 du descripteur de message (MQMD).

#### **Date d'insertion**

Date à laquelle un message a été inséré. La valeur initiale est une chaîne nulle.

#### **Heure d'insertion**

Heure à laquelle un message a été inséré. La valeur initiale est une chaîne nulle.

#### **nom du gestionnaire de files d'attente de réponse**

Nom du gestionnaire de files d'attente auquel toute réponse doit être envoyée. La valeur initiale est une chaîne nulle.

#### **Nom de file d'attente de réponses**

Nom de la file d'attente à laquelle toute réponse doit être envoyée. La valeur initiale est une chaîne nulle.

#### **rapport**

Informations de retour associées à un message. La valeur initiale est MQRO\_NONE. Les valeurs supplémentaires suivantes sont possibles:

- MQRO\_EXCEPTION
- MQRO\_EXCEPTION\_WITH\_DATA
- MQRO\_EXCEPTION\_WITH\_FULL\_DATA \*
- MQRO\_EXPIRATION
- MQRO\_EXPIRATION\_WITH\_DATA
- MQRO\_EXPIRATION\_AVEC\_DONNÉES\_COMPLET\*
- MQRO\_COA
- MQRO\_COA\_WITH\_DATA
- MQRO\_COA\_WITH\_FULL\_DATA \*
- MQRO\_COD
- MQRO\_COD\_WITH\_DATA
- MQRO\_COD\_WITH\_FULL\_DATA \*
- MQRO\_PAN
- MQRO\_NAN
- MQRO\_NEW\_MSG\_ID
- MQRO\_NEW\_CORREL\_ID
- MQRO\_COPY\_MSG\_ID\_TO\_CORREL\_ID
- MQRO\_PASS\_CORREL\_ID
- MQRO\_DEAD\_LETTER\_Q
- MQRO\_DISCARD\_MSG

où \* indique des valeurs qui ne sont pas prises en charge sous IBM MQ for z/OS.

#### **numéro de séquence**

Informations de séquence identifiant un message au sein d'un groupe. La valeur initiale est 1.

#### **longueur totale des messages**

Nombre d'octets disponibles lors de la dernière tentative de lecture d'un message. Ce nombre sera supérieur à la **longueur de message** ImqCache si le dernier message a été tronqué ou si le dernier message n'a pas été lu en raison d'une troncature. Cet attribut est en lecture seule. La valeur initiale est zéro.

Cet attribut peut être utile dans toute situation impliquant des messages tronqués.

### **ID utilisateur**

Identité d'utilisateur associée à un message. La valeur initiale est une chaîne nulle.

## **Constructeurs**

### **ImqMessage( );**

Constructeur par défaut.

### **ImqMessage( const ImqMessage & msg );**

Constructeur de copie. Voir la méthode **operator =** pour plus de détails.

## **Méthodes d'objet (public)**

### **void operator = ( const ImqMessage & msg );**

Copie le MQMD et les données de message à partir de *msg*. Si une mémoire tampon a été fournie par l'utilisateur pour cet objet, la quantité de données copiées est limitée à la taille de mémoire tampon disponible. Sinon, le système s'assure qu'une mémoire tampon de taille adéquate est mise à disposition pour les données copiées.

### **ImqString applicationIdDonnées () const ;**

Renvoie une copie des **données d'ID d'application**.

### **void setApplicationIdData ( const char \* data = 0 );**

Définit les **données d'ID application**.

### **ImqString applicationOriginDonnées () const ;**

Renvoie une copie des **données d'origine de l'application**.

### **void setApplicationOriginData ( const char \* data = 0 );**

Définit les **données d'origine de l'application**.

### **MQLONG backoutCount () const ;**

Renvoie le **nombre d'annulations**.

### **MQLONG characterSet () const ;**

Renvoie le **jeu de caractères**.

### **void setCharacterSet ( const MQLONG ccsid = MQCCSI\_Q\_MGR );**

Définit le **jeu de caractères**.

### **MQLONG codage () const ;**

Renvoie le **codage**.

### **void setEncoding ( const MQLONG encoding = MQENC\_NATIVE );**

Définit le **codage**.

### **MQLONG expiration () const ;**

Renvoie l' **expiration**.

### **void setExpiry ( const MQLONG expiration );**

Définit l' **expiration**.

### **ImqString format () const ;**

Renvoie une copie du **format**, y compris les blancs de fin.

### **ImqBoolean formatIs ( const char \* format-to-test ) const ;**

Renvoie TRUE si le **format** est identique à *format-to-test*.

### **void setFormat ( const char \* name = 0 );**

Définit le **format**, complété par huit caractères avec des blancs de fin.

### **MQLONG messageFlags () const ;**

Renvoie les **indicateurs de message**.

### **void setMessageFlags ( const MQLONG indicateurs );**

Définit les **indicateurs de message**.

**MQLONG messageType () const ;**  
Renvoie le **type de message**.

**void setMessageType ( const MQLONG type ) ;**  
Définit le **type de message**.

**MQLONG décalage () const ;**  
Renvoie le **décalage**.

**void setOffset ( const MQLONG décalage ) ;**  
Définit le **décalage**.

**MQLONG originalLength () const ;**  
Renvoie la **longueur d'origine**.

**void setOriginalLength ( const MQLONG longueur ) ;**  
Définit la **longueur d'origine**.

**MQLONG persistance () const ;**  
Renvoie la **persistance**.

**void setPersistence ( const MQLONG persistance ) ;**  
Définit la **persistance**.

**MQLONG priorité () const ;**  
Renvoie la **priorité**.

**void setPriority ( const MQLONG priorité ) ;**  
Définit la **priorité**.

**ImqString putApplicationNom () const ;**  
Renvoie une copie du **nom de l'application d'insertion**.

**void setPutApplicationName ( const char \* name = 0 ) ;**  
Définit le **nom de l'application d'insertion**.

**MQLONG putApplicationType () const ;**  
Renvoie le **type d'application d'insertion**.

**void setPutApplicationType ( const MQLONG type = MQAT\_NO\_CONTEXT ) ;**  
Définit le **type d'application d'insertion**.

**ImqString putDate () const ;**  
Renvoie une copie de la **date d'insertion**.

**void setPutDate ( const char \* date = 0 ) ;**  
Définit la **date d'insertion**.

**ImqString putTime () const ;**  
Renvoie une copie de l' **heure d'insertion**.

**void setPutTime ( const char \* time = 0 ) ;**  
Définit l' **heure d'insertion**.

**ImqBoolean readItem ( ImqItem & item ) ;**  
Lit dans l'objet *item* à partir de la mémoire tampon de messages, à l'aide de la méthode *ImqItem pasteIn* . Elle renvoie TRUE en cas de succès.

**ImqString replyToQueueManagerNom () const ;**  
Renvoie une copie du **nom du gestionnaire de files d'attente de réponse**.

**void setReplyToQueueManagerName ( const char \* name = 0 ) ;**  
Définit le **nom du gestionnaire de files d'attente de réponse**.

**ImqString replyToQueueName () const ;**  
Renvoie une copie du **nom de la file d'attente de réponse**.

**void setReplyToQueueNom ( const char \* name = 0 ) ;**  
Définit le **nom de la file d'attente de réponse**.

**MQLONG rapport () const ;**  
Renvoie le **rapport**.



**void setReport ( const MQLONG rapport ) ;**

Définit le **rapport**.

**MQLONG sequenceNumber () const ;**

Renvoie le **numéro de séquence**.

**void setSequenceNumber ( const MQLONG nombre ) ;**

Définit le **numéro de séquence**.

**size\_t totalMessageLongueur () const ;**

Renvoie la **longueur totale des messages**.

**ImqString userId () const ;**

Renvoie une copie de l' **ID utilisateur**.

**void setUserId ( const char \* id = 0 ) ;**

Définit l' **ID utilisateur**.

**ImqBoolean writeItem ( ImqItem & élément ) ;**

Écrit à partir de l'objet *item* dans la mémoire tampon de messages, à l'aide de la méthode *ImqItem copyOut* . L'écriture peut prendre la forme d'une insertion, d'un remplacement ou d'un ajout: cela dépend de la classe de l'objet *item* . Cette méthode renvoie TRUE en cas de succès.

### Méthodes d'objet (protégées)

**static void setVersionSupported ( const MQLONG ) ;**

Définit la **version MQMD**. La valeur par défaut est **MQMD\_VERSION\_2**.

### Données d'objet (protégées)

**z/OS** **MQMD1 omqmd**

Structure de données MQMD sous z/OS.

**Multi** **MQMD2 omqmd**

Structure de données MQMD sous Multiplateformes.

## Classe C++ de la fonction de suivi ImqMessage

Cette classe encapsule les attributs d'un objet *ImqMessage* ou *ImqQueue* qui peuvent être associés à l'un ou l'autre objet.

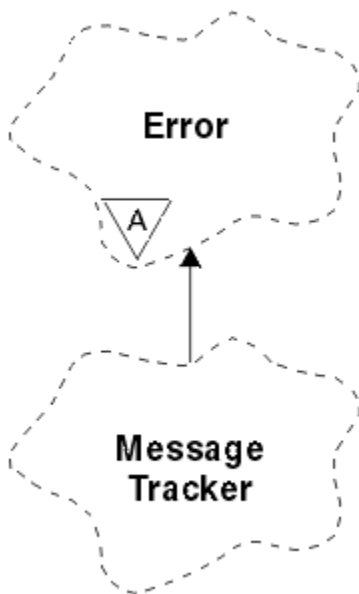


Figure 27. Classe *ImqMessageTracker*

Cette classe est liée aux appels MQI répertoriés dans «Référence croisée de la fonction de suivi ImqMessage», à la page 1877.

- «Attributs d'objet», à la page 1930
- «Constructeurs», à la page 1931
- «Méthodes d'objet (public)», à la page 1931
- «Codes raison», à la page 1932

## **Attributs d'objet**

### **jeton de comptabilité**

Valeur binaire (MQBYTE32) de longueur MQ\_ACCOUNTING\_TOKEN\_LENGTH. La valeur initiale est MQACT\_NONE.

### **ID de corrélation**

Une valeur binaire (MQBYTE24) de longueur MQ\_CORREL\_ID\_LENGTH que vous affectez à la corrélation des messages. La valeur initiale est MQCI\_NONE. La valeur supplémentaire, MQCI\_NEW\_SESSION, est possible.

### **commentaire**

Informations de retour à envoyer avec un message. La valeur initiale est MQFB\_NONE. Les valeurs supplémentaires suivantes sont possibles:

- MQFB\_SYSTEM\_FIRST
- MQFB\_SYSTEM\_LAST
- MQFB\_APPL\_FIRST
- MQFB\_APPL\_LAST
- MQFB\_COA
- MQFB\_COD
- EXPIRATION\_MQFB
- MQFB\_PAN
- MQFB\_NAN
- MQFB\_QUIT
- MQFB\_DATA\_LENGTH\_ZERO
- MQFB\_DONNÉES\_LONGUEUR\_NÉGATIVE
- MQFB\_DATA\_LENGTH\_TOO\_BIG
- MQFB\_BUFFER\_OVERFLOW
- MQFB\_LENGTH\_OFF\_BY\_ONE
- MQFB\_IIH\_ERREUR
- MQFB\_NOT\_AUTHORIZED\_FOR\_IMS
- MQFB\_IMS\_ERREUR
- MQFB\_IMS\_FIRST
- MQFB\_IMS\_LAST
- MQFB\_CICS\_APPL\_ABENDED
- MQFB\_CICS\_APPL\_NOT\_STARTED
- MQFB\_CICS\_BRIDGE\_FAILURE
- MQFB\_CICS\_CCSID\_ERREUR
- MQFB\_CICS\_CIH\_ERREUR
- MQFB\_CICS\_COMMAREA\_ERREUR
- MQFB\_CICS\_CORREL\_ID\_ERREUR

- MQFB\_CICS\_DLQ\_ERREUR
- MQFB\_CICS\_ENCODING\_ERROR
- MQFB\_CICS\_INTERNAL\_ERROR
- MQFB\_CICS\_NOT\_AUTHORIZED
- MQFB\_CICS\_UOW\_BACKED\_OUT
- MQFB\_ERREUR UOW\_CICS\_

Vous pouvez également utiliser une chaîne spécifique à l'application de votre choix. Pour plus d'informations à ce sujet, voir la zone «Commentaires en retour (MQLONG) pour MQMD», à la page 459 du descripteur de message (MQMD).

### **ID de groupe**

Valeur binaire (MQBYTE24) de longueur MQ\_GROUP\_ID\_LENGTH unique dans une file d'attente. La valeur initiale est MQGI\_NONE.

### **ID message**

Valeur binaire (MQBYTE24) de longueur MQ\_MSG\_ID\_LENGTH unique dans une file d'attente. La valeur initiale est MQMI\_NONE.

## **Constructeurs**

### **ImqMessageTracker () ;**

Constructeur par défaut.

### **ImqMessageTracker ( const ImqMessageTracker & tracker ) ;**

Constructeur de copie. Voir la méthode **operator =** pour plus de détails.

## **Méthodes d'objet (public)**

### **void operator = ( const ImqMessageTracker & tracker ) ;**

Copie les données d'instance à partir du *dispositif de suivi*, en remplaçant les données d'instance existantes.

### **ImqBinary accountingToken () const ;**

Renvoie une copie du **jeton de comptabilité**.

### **ImqBoolean setAccountingToken ( const ImqBinary & jeton ) ;**

Définit le **jeton de comptabilité**. La **longueur de données** du *jeton* doit être égale à zéro ou MQ\_ACCOUNTING\_TOKEN\_LENGTH. Cette méthode renvoie TRUE en cas de succès.

### **void setAccountingToken ( const MQBYTE32 token = 0 ) ;**

Définit le **jeton de comptabilité**. La valeur de *token* peut être zéro, ce qui revient à spécifier MQACT\_NONE. Si *token* est différent de zéro, il doit traiter les octets MQ\_ACCOUNTING\_TOKEN\_LENGTH des données binaires. Lorsque vous utilisez des valeurs prédéfinies telles que MQACT\_NONE, vous pouvez être amené à effectuer un transtypage pour garantir une correspondance de signature ; par exemple, (MQBYTE \*) MQACT\_NONE.

### **ImqBinary correlationId () const ;**

Renvoie une copie de l' **ID corrélation**.

### **ImqBoolean setCorrelationId ( const ImqBinary & jeton ) ;**

Définit l' **ID de corrélation**. La **longueur de données** du *jeton* doit être égale à zéro ou MQ\_CORREL\_ID\_LENGTH. Cette méthode renvoie TRUE en cas de succès.

### **void setCorrelationId ( const MQBYTE24 id = 0 ) ;**

Définit l' **ID de corrélation**. *id* peut être égal à zéro, ce qui revient à spécifier MQCI\_NONE. Si *id* est différent de zéro, il doit traiter les octets MQ\_CORREL\_ID\_LENGTH des données binaires. Lorsque vous utilisez des valeurs prédéfinies telles que MQCI\_NONE, vous pouvez être amené à effectuer un transtypage pour garantir une correspondance de signature ; par exemple, (MQBYTE \*) MQCI\_NONE.

### **commentaires en retour MQLONG () const ;**

Renvoie les **commentaires en retour**.

**void setFeedback ( const MQLONG feedback ) ;**

Définit les **commentaires en retour**.

**ImqBinary groupId () const ;**

Renvoie une copie de l' **ID groupe**.

**ImqBoolean setGroupId ( const ImqBinary & jeton ) ;**

Définit l' **ID groupe**. La **longueur des données** du *jeton* doit être zéro ou MQ\_GROUP\_ID\_LENGTH. Cette méthode renvoie TRUE en cas de succès.

**void setGroupId ( const MQBYTE24 id = 0 ) ;**

Définit l' **ID groupe**. *id* peut être égal à zéro, ce qui revient à spécifier MQGI\_NONE. Si *id* est différent de zéro, il doit traiter les octets MQ\_GROUP\_ID\_LENGTH des données binaires. Lorsque vous utilisez des valeurs prédéfinies telles que MQGI\_NONE, vous devrez peut-être effectuer un transtypage pour garantir une correspondance de signature, par exemple (MQBYTE \*) MQGI\_NONE.

**ImqBinary messageId () const ;**

Renvoie une copie de l' **ID message**.

**ImqBoolean setMessageId ( const ImqBinary & token ) ;**

Définit l' **ID message**. La **longueur de données** du *jeton* doit être zéro ou MQ\_MSG\_ID\_LENGTH. Cette méthode renvoie TRUE en cas de succès.

**void setMessageId ( const MQBYTE24 id = 0 ) ;**

Définit l' **ID message**. *id* peut être égal à zéro, ce qui revient à spécifier MQMI\_NONE. Si *id* est différent de zéro, il doit traiter les octets MQ\_MSG\_ID\_LENGTH des données binaires. Lorsque vous utilisez des valeurs prédéfinies telles que MQMI\_NONE, vous pouvez être amené à effectuer un transtypage pour garantir une correspondance de signature, par exemple (MQBYTE \*) MQMI\_NONE.

## Codes raison

- MQRC\_BINARY\_DATA\_LENGTH\_ERROR

## Classe C++ ImqNamelist

Cette classe encapsule une liste de noms.

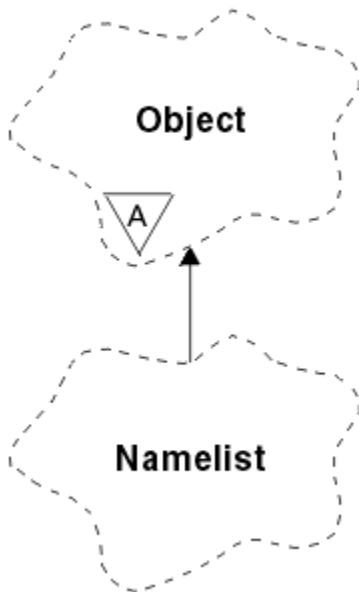


Figure 28. Classe `ImqNamelist`

Cette classe est liée aux appels MQI répertoriés dans [«Référence croisée ImqNamelist»](#), à la page 1877.

- [«Attributs d'objet»](#), à la page 1933
- [«Constructeurs»](#), à la page 1933

- «Méthodes d'objet (public)», à la page 1933
- «Codes raison», à la page 1933

## Attributs d'objet

### Nombre de noms

Nombre de noms d'objet dans les **noms de liste de noms**. Cet attribut est en lecture seule.

### noms de liste de noms

Noms d'objet, dont le nombre est indiqué par le **nombre de noms**. Cet attribut est en lecture seule.

## Constructeurs

### ImqNamelist();

Constructeur par défaut.

### ImqNamelist(const ImqNamelist & liste );

Constructeur de copie. Le **statut d'ouverture** ImqObject est false.

### ImqNamelist(const char \* nom );

Définit le nom ImqObject sur **name**.

## Méthodes d'objet (public)

### void operator = (const ImqNamelist & liste );

Copie les données d'instance à partir de la *liste*, en remplaçant les données d'instance existantes. Le **statut d'ouverture** ImqObject est false.

### ImqBoolean nameCount(MQLONG & nombre );

Fournit une copie du **nombre de noms**. Elle renvoie TRUE en cas de succès.

### MQLONG nameCount ();

Renvoie le **nombre de noms** sans aucune indication d'erreurs possibles.

### ImqBoolean namelistName (const MQLONG index, ImqString & nom );

Fournit une copie de l'un des **noms de liste de noms** par index à base de zéro. Elle renvoie TRUE en cas de succès.

### ImqString namelistName (const MQLONG index );

Renvoie l'un des **noms de liste de noms** par index à base de zéro sans aucune indication d'erreurs possibles.

## Codes raison

- ERREUR MQRC\_INDEX\_ERREUR
- MQRC\_INDEX\_NOT\_PRESENT

## Classe C++ ImqObject

Cette classe est abstraite. Lorsqu'un objet de cette classe est détruit, il est automatiquement fermé et sa connexion au gestionnaire ImqQueueest interrompue.

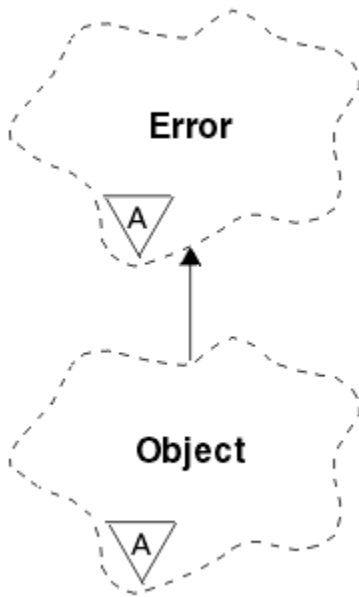


Figure 29. Classe *ImqObject*

Cette classe est liée aux appels MQI répertoriés dans [«Référence croisée ImqObject»](#), à la page 1878.

- [«Attributs de classe»](#), à la page 1934
- [«Attributs d'objet»](#), à la page 1934
- [«Constructeurs»](#), à la page 1936
- [«Méthodes de classe \(public\)»](#), à la page 1936
- [«Méthodes d'objet \(public\)»](#), à la page 1936
- [«Méthodes d'objet \(protégées\)»](#), à la page 1938
- [«Données d'objet \(protégées\)»](#), à la page 1939
- [«Codes raison»](#), à la page 1939
- 

## Attributs de classe

### comportement

Contrôle le comportement de l'ouverture implicite.

#### **IMQ\_IMPL\_OPEN (8L)**

L'ouverture implicite est autorisée. Il s'agit de l'option par défaut.

## Attributs d'objet

### Date de modification

Date de modification. Cet attribut est en lecture seule.

### Heure de modification

Heure de modification. Cet attribut est en lecture seule.

### Autre ID utilisateur

ID utilisateur de remplacement, jusqu'à MQ\_USER\_ID\_LENGTH caractères. La valeur initiale est une chaîne nulle.

### Autre ID de sécurité

ID de sécurité de remplacement. Une valeur binaire (MQBYTE40) de longueur MQ\_SECURITY\_ID\_LENGTH. La valeur initiale est MQSID\_NONE.

### options de fermeture

Options qui s'appliquent lorsqu'un objet est fermé. La valeur initiale est MQCO\_NONE. Cet attribut est ignoré lors des opérations de réouverture implicites, où la valeur MQCO\_NONE est toujours utilisée.

### référence de connexion

Référence à un objet gestionnaire ImqQueue qui fournit la connexion requise à un gestionnaire de files d'attente (local). Pour un objet gestionnaire ImqQueue, il s'agit de l'objet lui-même. La valeur initiale est zéro.

**Remarque :** Ne le confondez pas avec le nom du gestionnaire de files d'attente qui identifie un gestionnaire de files d'attente (éventuellement distant) pour une file d'attente nommée.

### description

Nom descriptif (jusqu'à 64 caractères) du gestionnaire de files d'attente, de la file d'attente, de la liste de noms ou du processus. Cet attribut est en lecture seule.

### Nom

Nom (jusqu'à 48 caractères) du gestionnaire de files d'attente, de la file d'attente, de la liste de noms ou du processus. La valeur initiale est une chaîne nulle. Le nom d'une file d'attente modèle change après un **open** et devient le nom de la file d'attente dynamique résultante.

**Remarque :** Un gestionnaire ImqQueue peut avoir un nom null, représentant le gestionnaire de files d'attente par défaut. Le nom change pour le gestionnaire de files d'attente réel après une ouverture réussie. Une liste ImqDistribution est dynamique et doit avoir un nom null.

### objet géré suivant

Il s'agit de l'objet suivant de cette classe, sans ordre particulier, ayant la même référence de connexion que cet objet. La valeur initiale est zéro.

### Options d'ouverture

Options qui s'appliquent lorsqu'un objet est ouvert. La valeur initiale est MQOO\_INQUIRE. Il existe deux façons de définir les valeurs appropriées:

1. Ne définissez pas les options d'ouverture et n'utilisez pas la méthode d'ouverture. IBM MQ ajuste automatiquement les options d'ouverture et ouvre, rouvre et ferme automatiquement les objets selon les besoins. Cela peut entraîner des opérations de réouverture inutiles, car IBM MQ utilise la méthode `openFor` et ajoute des options d'ouverture de manière incrémentielle uniquement.
2. Définissez les options d'ouverture avant d'utiliser les méthodes qui génèrent un appel MQI (voir «[Références croisées C++ et MQI](#)», à la [page 1871](#)). Cela permet de s'assurer que des opérations de réouverture inutiles ne se produisent pas. Définissez les options d'ouverture de manière explicite si l'un des problèmes de réouverture potentiels est susceptible de se produire (voir [Rouvrir](#)).

Si vous utilisez la méthode d'ouverture, vous devez d'abord vous assurer que les options d'ouverture sont appropriées. Toutefois, l'utilisation de la méthode ouverte n'est pas obligatoire ; IBM MQ présente toujours le même comportement que dans le cas 1, mais dans ce cas, le comportement est efficace.

Zéro n'est pas une valeur valide ; définissez la valeur appropriée avant de tenter d'ouvrir l'objet. Pour ce faire, utilisez soit **setOpenOptions** (*IOpenOptions*) suivi de **open** (), soit **openFor** (*IRequiredOpenOption*).

### Remarque :

1. MQOO\_OUTPUT est remplacé par MQOO\_INQUIRE lors de la méthode **open** pour une liste de distribution, car MQOO\_OUTPUT est le seul **open option** valide pour le moment. Toutefois, il est recommandé de toujours définir MQOO\_OUTPUT explicitement dans les programmes d'application qui utilisent la méthode **open**.
2. Spécifiez MQOO\_RESOLVE\_NAMES si vous souhaitez utiliser les attributs **resolved queue manager name** et **resolved queue name** de la classe.

### état ouvert

Indique si l'objet est ouvert (TRUE) ou fermé (FALSE). La valeur initiale est FALSE. Cet attribut est en lecture seule.

**objet géré précédent**

L'objet précédent de cette classe, sans ordre particulier, ayant la même référence de connexion que cet objet. La valeur initiale est zéro.

**identificateur-gestionnaire de files d'attente**

Identificateur du gestionnaire de files d'attente. Cet attribut est en lecture seule.

**Constructeurs****ImqObject();**

Constructeur par défaut.

**ImqObject(const ImqObject & objet );**

Constructeur de copie. Le statut d'ouverture sera FALSE.

**Méthodes de classe (public)****comportement MQLONG statique ();**

Renvoie le comportement.

**void setBehavior(const MQLONG behavior = 0);**

Définit le comportement.

**Méthodes d'objet (public)****void operator = (const ImqObject & objet );**

Effectue une fermeture si nécessaire et copie les données d'instance à partir de l'objet . Le statut d'ouverture sera FALSE.

**ImqBoolean alterationDate( ImqString & date );**

Fournit une copie de la date de modification. Elle renvoie TRUE en cas de succès.

**ImqString alterationDate();**

Renvoie la date de modification sans aucune indication d'erreurs possibles.

**ImqBoolean alterationTime( ImqString & heure );**

Fournit une copie de l'heure de modification. Elle renvoie TRUE en cas de succès.

**ImqString alterationTime();**

Renvoie l'heure de modification sans aucune indication d'erreurs possibles.

**ID ImqString alternateUser() const ;**

Renvoie une copie de l'ID utilisateur de remplacement.

**ImqBoolean setAlternateUserId (const char \* id );**

Définit l'ID utilisateur de remplacement. L'ID utilisateur de remplacement peut être défini uniquement lorsque le statut d'ouverture est FALSE. Cette méthode renvoie TRUE en cas de succès.

**ID ImqBinary alternateSecurity() const ;**

Renvoie une copie de l'autre ID de sécurité.

**ImqBoolean setAlternateSecurityId(const ImqBinary & jeton );**

Définit l'autre ID de sécurité. L'ID de sécurité de remplacement peut être défini uniquement lorsque le statut d'ouverture est FALSE. La longueur des données du *jeton* doit être zéro ou MQ\_SECURITY\_ID\_LENGTH. Elle renvoie TRUE en cas de succès.

**ImqBoolean setAlternateSecurityId(const MQBYTE\* token = 0);**

Définit l'autre ID de sécurité. La valeur de *token* peut être égale à zéro, ce qui revient à spécifier MQSID\_NONE. Si *token* est différent de zéro, il doit traiter les octets MQ\_SECURITY\_ID\_LENGTH des données binaires. Lorsque vous utilisez des valeurs prédéfinies telles que MQSID\_NONE, vous pouvez être amené à effectuer un transtypage pour garantir la correspondance de signature ; par exemple, (MQBYTE \*) MQSID\_NONE.

L'ID de sécurité de remplacement peut être défini uniquement lorsque le statut d'ouverture est TRUE. Elle renvoie TRUE en cas de succès.



**ImqBoolean setAlternateSecurityId(const unsigned char \* id = 0) ;**

Définit l'autre ID de sécurité.

**ImqBoolean close () ;**

Définit le statut d'ouverture sur FALSE. Elle renvoie TRUE en cas de succès.

**MQLONG closeOptions () const ;**

Renvoie les options de fermeture.

**void setCloseOptions (const MQLONG options) ;**

Définit les options de fermeture.

**ImqQueueManager \* connectionReference () const ;**

Renvoie la référence de connexion.

**void setConnectionReference ( ImqQueueManager & manager ) ;**

Définit la référence de connexion.

**void setConnectionReference ( ImqQueueManager \* manager = 0 ) ;**

Définit la référence de connexion.

**description ImqBoolean virtuelle ( ImqString & description ) = 0 ;**

Fournit une copie de la description. Elle renvoie TRUE en cas de succès.

**Description d' ImqString () ;**

Renvoie une copie de la description sans indication d'erreurs possibles.

**nom ImqBoolean virtuel ( ImqString & nom ) ;**

Fournit une copie du nom. Elle renvoie TRUE en cas de succès.

**ImqString name () ;**

Renvoie une copie du nom sans indication des erreurs possibles.

**ImqBoolean setName (const car \* name = 0) ;**

Définit le nom. Le nom peut être défini uniquement lorsque le statut d'ouverture est FALSE et, pour un gestionnaire ImqQueue, lorsque le statut de connexion est FALSE. Elle renvoie TRUE en cas de succès.

**ImqObject \* nextManagedObject () const ;**

Renvoie l'objet géré suivant.

**ImqBoolean open () ;**

Remplace le statut d'ouverture par TRUE en ouvrant l'objet si nécessaire, en utilisant, entre autres attributs, les options d'ouverture et le nom. Cette méthode utilise les informations de référence de connexion et la méthode de connexion du gestionnaire ImqQueue si nécessaire pour s'assurer que le statut de connexion du gestionnaire ImqQueue est TRUE. Elle renvoie le statut ouvert.

**ImqBoolean openFor (const MQLONG required-options = 0) ;**

Tente de s'assurer que l'objet est ouvert avec des options d'ouverture ou avec des options d'ouverture qui garantissent le comportement impliqué par la valeur du paramètre *required-options* .

Si *required-options* est égal à zéro, une entrée est requise et toute option d'entrée suffit. Par conséquent, si les options d'ouverture contiennent déjà l'une des options suivantes:

- MQOO\_INPUT\_AS\_Q\_DEF
- MQOO\_INPUT\_SHARED
- MQOO\_INPUT\_EXCLUSIVE

les options d'ouverture sont déjà satisfaisantes et ne sont pas modifiées ; si les options d'ouverture ne contiennent aucune de ces options, MQOO\_INPUT\_AS\_Q\_DEF est défini dans les options d'ouverture.

Si *required-options* est différent de zéro, les options requises sont ajoutées aux options d'ouverture ; si *required-options* est l'une de ces options, les autres sont réinitialisées.

Si l'une des options d'ouverture est modifiée et que l'objet est déjà ouvert, l'objet est fermé temporairement et rouvert afin d'ajuster les options d'ouverture.

Elle renvoie TRUE en cas de succès. La réussite indique que l'objet est ouvert avec les options appropriées.

**MQLONG openOptions () const ;**

Renvoie les options d'ouverture.

**ImqBoolean setOpenOptions (const MQLONG options ) ;**

Définit les options d'ouverture. Les options d'ouverture ne peuvent être définies que lorsque le statut d'ouverture est FALSE. Elle renvoie TRUE en cas de succès.

**ImqBoolean openStatus () const ;**

Renvoie le statut d'ouverture.

**ImqObject \* previousManagedObject () const ;**

Renvoie l'objet géré précédent.

**ImqBoolean queueManagerIdentificateur ( ImqString & id ) ;**

Fournit une copie de l'identificateur du gestionnaire de files d'attente. Elle renvoie TRUE en cas de succès.

**ImqString queueManagerIdentificateur () ;**

Renvoie l'identificateur du gestionnaire de files d'attente sans aucune indication d'erreurs possibles.

**Méthodes d'objet (protégées)****virtual ImqBoolean closeTemporarily () ;**

Ferme un objet en toute sécurité avant de le rouvrir. Elle renvoie TRUE en cas de succès. Cette méthode suppose que le statut d'ouverture est TRUE.

**MQHCONN connectionHandle () const ;**

Renvoie le MQHCONN associé à la référence de connexion. Cette valeur est égale à zéro s'il n'existe aucune référence de connexion ou si le gestionnaire n'est pas connecté.

**ImqBoolean inquire (const MQLONG int-attr, MQLONG & valeur ) ;**

Renvoie une valeur entière dont l'index est une valeur MQIA\_\*. En cas d'erreur, la valeur est définie sur MQIAV\_UNDEFINED.

**ImqBoolean inquire (const MQLONG char-attr, char \* & buffer, const size\_t longueur ) ;**

Renvoie une chaîne de caractères dont l'index est une valeur MQCA\_\*.

**Remarque :** Ces deux méthodes ne renvoient qu'une seule valeur d'attribut. Si une *image instantanée* est requise de plusieurs valeurs, où les valeurs sont cohérentes les unes avec les autres pendant un instant, IBM MQ C++ ne fournit pas cette fonction et vous devez utiliser l'appel MQINQ avec les paramètres appropriés.

**vide virtuel openInformationDisperse () ;**

Disperse les informations de la section variable de la structure de données MQOD immédiatement après un appel MQOPEN.

**virtuel ImqBoolean openInformationPrepare () ;**

Prépare les informations pour la section variable de la structure de données MQOD immédiatement avant un appel MQOPEN et renvoie TRUE si l'opération aboutit.

**ImqBoolean set (const MQLONG int-attr, const MQLONG valeur ) ;**

Définit un attribut entier IBM MQ .

**ImqBoolean set (const MQLONG char-attr, const char \* buffer, const size\_t required-length ) ;**

Définit un attribut de caractère IBM MQ .

**void setNextManagedObject (const ImqObject \* object = 0) ;**

Définit l'objet géré suivant.

Attention: Utilisez cette fonction uniquement si vous êtes sûr qu'elle ne rompt pas la liste des objets gérés.

**void setPreviousManagedObject (const ImqObject \* object = 0) ;**

Définit l'objet géré précédent.

Attention: Utilisez cette fonction uniquement si vous êtes sûr qu'elle ne rompt pas la liste des objets gérés.

## Données d'objet (protégées)

### **MQHOBJ** *ohobj*

Descripteur d'objet IBM MQ (valide uniquement lorsque le statut d'ouverture est TRUE).

### **MQOD** *capsule*

Structure de données MQOD imbriquée. La quantité de mémoire allouée à cette structure de données est celle requise pour un MQOD version 2. Inspectez le numéro de version (*omqod.Version*) et accédez aux autres zones comme suit:

#### **MQOD\_VERSION\_1**

Toutes les autres zones de *omqod* sont accessibles.

#### **MQOD\_VERSION\_2**

Toutes les autres zones de *omqod* sont accessibles.

#### **MQOD\_VERSION\_3**

*omqod.pmqod* est un pointeur vers un MQOD alloué de manière dynamique, plus grand.

Aucune autre zone de *omqod* n'est accessible. Toutes les zones traitées par *omqod.pmqod* sont accessibles.

**Remarque :** *omqod.pmqod.Version* peut être inférieur à *omqod.Version*, ce qui indique que IBM MQ MQI client possède plus de fonctionnalités que le serveur IBM MQ .

## Codes raison

- MQRC\_ATTRIBUT\_LOCKED
- MQRC\_INCONSISTENT\_OBJECT\_STATE
- MQRC\_NO RÉFÉRENCE\_CONNEXION
- MQRC\_STORAGE\_NOT\_AVAILABLE
- MQRC\_REOPEN\_SAVED\_CONTEXT\_ERR
- (codes anomalie de MQCLOSE)
- (codes anomalie de MQCONN)
- (codes anomalie de MQINQ)
- (codes anomalie de MQOPEN)
- (codes anomalie de MQSET)

## Classe C++ ImqProcess

Cette classe encapsule un processus d'application (objet IBM MQ de type MQOT\_PROCESS) qui peut être déclenché par un moniteur de déclenchement.

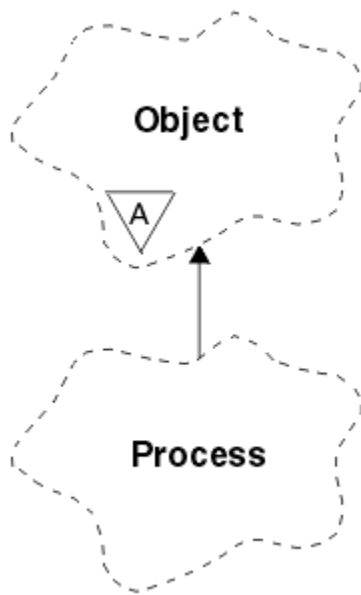


Figure 30. Classe *ImqProcess*

- «Attributs d'objet», à la page 1940
- «Constructeurs», à la page 1940
- «Méthodes d'objet (public)», à la page 1940

## Attributs d'objet

### ID application

Identité du processus d'application. Cet attribut est en lecture seule.

### type d'application

Type du processus d'application. Cet attribut est en lecture seule.

### données d'environnement

Informations d'environnement pour le processus. Cet attribut est en lecture seule.

### données utilisateur

Données utilisateur du processus. Cet attribut est en lecture seule.

## Constructeurs

### **ImqProcess();**

Constructeur par défaut.

### **ImqProcess( const ImqProcess & process );**

Constructeur de copie. L' **état d'ouverture** ImqObject est FALSE.

### **ImqProcess( const char \* nom );**

Définit le **nom** ImqObject .

## Méthodes d'objet (public)

### **void operator = ( const ImqProcess & process );**

Effectue une opération de fermeture si nécessaire, puis copie les données d'instance à partir du *processus*. L' **état d'ouverture** ImqObject est FALSE.

### **ImqBoolean applicationId ( ImqString & id );**

Fournit une copie de l' **ID application**. Elle renvoie TRUE en cas de succès.

### **ImqString applicationId ( );**

Renvoie l' **ID application** sans aucune indication d'erreurs possibles.

**ImqBoolean applicationType ( MQLONG & type ) ;**

Fournit une copie du **type d'application**. Elle renvoie TRUE en cas de succès.

**MQLONG applicationType () ;**

Renvoie le **type d'application** sans aucune indication d'erreurs possibles.

**ImqBoolean environmentData ( ImqString & données ) ;**

Fournit une copie des **données d'environnement**. Elle renvoie TRUE en cas de succès.

**ImqString environmentData () ;**

Renvoie les **données d'environnement** sans indication des erreurs possibles.

**ImqBoolean userData ( ImqString & données ) ;**

Fournit une copie des **données utilisateur**. Elle renvoie TRUE en cas de succès.

**ImqString userData () ;**

Renvoie les **données utilisateur** sans aucune indication d'erreurs possibles.

## Classe C++ ImqPutMessageOptions

Cette classe encapsule la structure de données MQPMO.

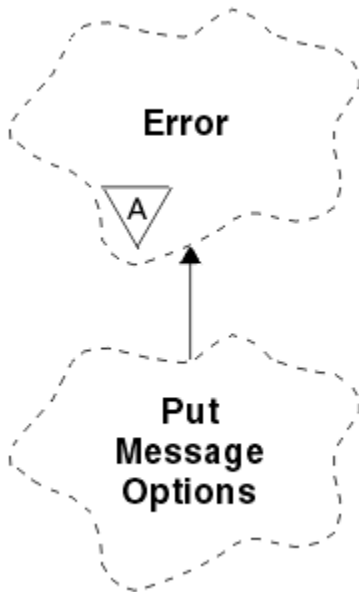


Figure 31. Classe `ImqPutMessageOptions`

- «Attributs d'objet», à la page 1941
- «Constructeurs», à la page 1942
- «Méthodes d'objet (public)», à la page 1942
- «Données d'objet (protégées)», à la page 1943
- «Codes raison», à la page 1943

### Attributs d'objet

#### référence de contexte

Une file d'attente `ImqQueue` qui fournit un contexte pour les messages. Au départ, il n'y a pas de référence.

#### options

Options du message d'insertion. La valeur initiale est `MQPMO_NONE`. Les valeurs supplémentaires suivantes sont possibles:

- `MQPMO_SYNCPOINT`
- `MQPMO_NO_SYNCPOINT`

- ID MQPMO\_NEW\_MSG\_ID
- MQPMO\_NEW\_CORREL\_ID
- MQPMO\_LOGICAL\_ORDER
- MQPMO\_NO\_CONTEXT
- MQPMO\_CONTEXTE\_PAR\_DÉFAUT
- MQPMO\_PASS\_IDENTITY\_CONTEXT
- MQPMO\_PASS\_ALL\_CONTEXT
- MQPMO\_SET\_IDENTITY\_CONTEXT
- MQPMO\_SET\_ALL\_CONTEXT
- MQPMO\_ALTERNATE\_USER\_AUTHORITY
- MQPMO\_FAIL\_IF QUIESCING

### zones d'enregistrement

Indicateurs qui contrôlent l'inclusion des enregistrements de message d'insertion lorsqu'un message est inséré. La valeur initiale est MQPMRF\_NONE. Les valeurs supplémentaires suivantes sont possibles:

- ID MQPMRF\_MSG\_ID
- ID\_CORREL\_MQPMRF\_
- ID\_GROUPE\_MQPMRF\_ID
- MQPMRF\_FEEDBACK
- MQPMRF\_COMPTING\_TOKEN

Les attributs de la fonction de suivi ImqMessagesont extraits de l'objet pour toute zone spécifiée. Les attributs de la fonction de suivi ImqMessagesont extraits de l'objet ImqMessage pour toute zone qui n'est pas spécifiée.

### Nom de gestionnaire de files d'attente résolu

Nom d'un gestionnaire de files d'attente de destination déterminé lors d'une insertion. La valeur initiale est null. Cet attribut est en lecture seule.

### Nom de file d'attente résolu

Nom d'une file d'attente de destination déterminée lors d'une insertion. La valeur initiale est null. Cet attribut est en lecture seule.

### participation au point de synchronisation

TRUE lorsque les messages sont placés sous contrôle de point de synchronisation.

## Constructeurs

### **ImqPutMessageOptions( );**

Constructeur par défaut.

### **ImqPutMessageOptions(const ImqPutMessageOptions & pmo );**

Constructeur de copie.

## Méthodes d'objet (public)

### **void operator = (const ImqPutMessageOptions & pmo );**

Copie les données d'instance à partir de *pmo*, en remplaçant les données d'instance existantes.

### **ImqQueue \* contextReference () const ;**

Renvoie la référence de contexte.

### **void setContextReference (const ImqQueue & file d'attente );**

Définit la référence de contexte.

### **void setContextReference (const ImqQueue \* queue = 0 );**

Définit la référence de contexte.

**Options MQLONG () const ;**

Renvoie les options.

**void setOptions (const MQLONG options ) ;**

Définit les options, y compris la valeur de participation du point de synchronisation.

**MQLONG recordFields () const ;**

Renvoie les zones d'enregistrement.

**void setRecordFields (const MQLONG fields ) ;**

Définit les zones d'enregistrement.

**ImqString resolvedQueueManagerName () const ;**

Renvoie une copie du nom du gestionnaire de files d'attente résolu.

**ImqString resolvedQueueNom () const ;**

Renvoie une copie du nom de la file d'attente résolue.

**ImqBoolean syncPointParticipation () const ;**

Renvoie la valeur de participation de point de synchronisation, qui est TRUE si les options incluent MQPMO\_SYNCPOINT.

**void setSyncPointParticipation (const ImqBoolean sync ) ;**

Définit la valeur de participation du point de synchronisation. Si *sync* est TRUE, les options sont modifiées pour inclure MQPMO\_SYNCPOINT et exclure MQPMO\_NO\_SYNCPOINT. Si *sync* est FALSE, les options sont modifiées pour inclure MQPMO\_NO\_SYNCPOINT et exclure MQPMO\_SYNCPOINT.

**Données d'objet (protégées)****omqpmo MQPMO**

Structure de données MQPMO.

**Codes raison**

- MQRC\_STORAGE\_NOT\_AVAILABLE

**Classe C++ ImqQueue**

Cette classe encapsule une file d'attente de messages (objet IBM MQ de type MQOT\_Q).

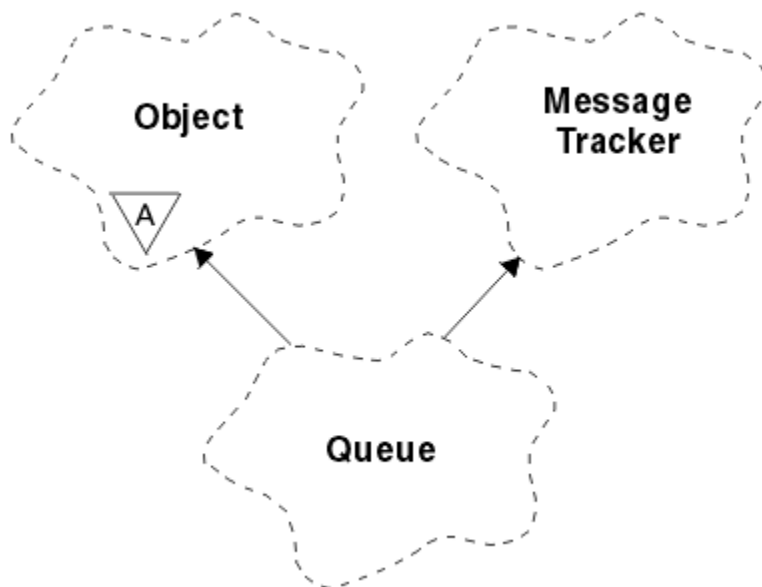


Figure 32. Classe ImqQueue

Cette classe est liée aux appels MQI répertoriés dans [Tableau 862](#), à la page 1879.

- [«Attributs d'objet», à la page 1944](#)
- [«Constructeurs», à la page 1947](#)
- [«Méthodes d'objet \(public\)», à la page 1947](#)
- [«Méthodes d'objet \(protégées\)», à la page 1953](#)
- [«Codes raison», à la page 1954](#)

## **Attributs d'objet**

### **Nom de file d'attente d'annulation**

Nom de remise en file d'attente d'annulation trop élevé. Cet attribut est en lecture seule.

### **seuil d'annulation**

Seuil d'annulation. Cet attribut est en lecture seule.

### **nom file d'attente de base**

Nom de la file d'attente dans laquelle l'alias est résolu. Cet attribut est en lecture seule.

### **nom de cluster**

Nom de cluster. Cet attribut est en lecture seule.

### **Liste de noms du cluster**

Nom de la liste de noms de cluster. Cet attribut est en lecture seule.

### **Rang charge cluster**

Rang charge cluster Cet attribut est en lecture seule.

### **Priorité charge cluster**

Priorité charge cluster Cet attribut est en lecture seule.

### **File d'attente d'utilisation de charge de travail de cluster**

Valeur de file d'attente d'utilisation de charge de travail de cluster Cet attribut est en lecture seule.

### **date de création**

Données de création de file d'attente. Cet attribut est en lecture seule.

### **heure de création**

Heure de création de la file d'attente. Cet attribut est en lecture seule.

### **Longueur en cours**

Nombre de messages dans la file d'attente. Cet attribut est en lecture seule.

### **liaison par défaut**

Liaison par défaut. Cet attribut est en lecture seule.

### **Option d'ouverture en entrée par défaut**

Option open-for-input par défaut. Cet attribut est en lecture seule.

### **Persistance par défaut**

Persistance de message par défaut. Cet attribut est en lecture seule.

### **Priorité par défaut**

Priorité de message par défaut. Cet attribut est en lecture seule.

### **type de définition**

Type de définition de file d'attente. Cet attribut est en lecture seule.

### **événement de profondeur élevée**

Attribut de contrôle pour les événements de longueur élevée de file d'attente. Cet attribut est en lecture seule.

### **limite supérieure de la profondeur**

Limite supérieure de la longueur de la file d'attente. Cet attribut est en lecture seule.

### **événement de profondeur faible**

Attribut de contrôle pour les événements de longueur faible de la file d'attente. Cet attribut est en lecture seule.

### **limite inférieure de la profondeur**

Limite inférieure de la longueur de la file d'attente. Cet attribut est en lecture seule.



**Événement de profondeur maximale**

Attribut de contrôle pour les événements de longueur maximale de la file d'attente. Cet attribut est en lecture seule.

**référence de liste de distribution**

Référence facultative à une liste ImqDistribution qui peut être utilisée pour distribuer des messages à plusieurs files d'attente, y compris celle-ci. La valeur initiale est null.

**Remarque :** Lorsqu'un objet ImqQueue est ouvert, tout objet de liste ImqDistribution ouvert auquel il fait référence est automatiquement fermé.

**listes de distribution**

Capacité d'une file d'attente de transmission à prendre en charge les listes de distribution. Cet attribut est en lecture seule.

**Nom de la file d'attente dynamique**

Nom de la file d'attente dynamique. La valeur initiale est AMQ.\* pour toutes les plateformes AIX, Linux, and Windows .

**Sauvegarde nb d'annulations**

Indique s'il faut renforcer le nombre d'annulations. Cet attribut est en lecture seule.

**Type d'index**

Type d'index. Cet attribut est en lecture seule.

**empêcher l'obtention**

Indique si les opérations d'extraction sont autorisées. La valeur initiale dépend de la définition de la file d'attente. Cet attribut est valide uniquement pour un alias ou une file d'attente locale.

**interdire l'insertion**

Indique si les opérations d'insertion sont autorisées. La valeur initiale dépend de la définition de la file d'attente.

**Nom de la file d'attente d'initialisation**

Nom de la file d'attente d'initialisation. Cet attribut est en lecture seule.

**profondeur maximale**

Nombre maximal de messages admis dans la file d'attente. Cet attribut est en lecture seule.

**longueur maximale des messages**

Longueur maximale de tout message de cette file d'attente, qui peut être inférieure à la longueur maximale de toute file d'attente gérée par le gestionnaire de files d'attente associé. Cet attribut est en lecture seule.

**Séquence livraison messages**

Indique si la priorité du message est pertinente. Cet attribut est en lecture seule.

**file d'attente répartie suivante**

Objet suivant de cette classe, sans ordre particulier, ayant la même **référence de liste de distribution** que cet objet. La valeur initiale est zéro.

Si un objet d'une chaîne est supprimé, l'objet précédent et l'objet suivant sont mis à jour de sorte que leurs liens de file d'attente répartie ne pointent plus vers l'objet supprimé.

**classe de message non persistant**

Niveau de fiabilité des messages non persistants insérés dans cette file d'attente. Cet attribut est en lecture seule.

**Nb d'ouvertures en entrée**

Nombre d'objets ImqQueue ouverts en entrée. Cet attribut est en lecture seule.

**Nb d'ouvertures en sortie**

Nombre d'objets ImqQueue ouverts pour la sortie. Cet attribut est en lecture seule.

**file d'attente distribuée précédente**

Objet précédent de cette classe, sans ordre particulier, ayant la même **référence de liste de distribution** que cet objet. La valeur initiale est zéro.

Si un objet d'une chaîne est supprimé, l'objet précédent et l'objet suivant sont mis à jour de sorte que leurs liens de file d'attente répartie ne pointent plus vers l'objet supprimé.

**nom de processus**

Nom de la définition de processus. Cet attribut est en lecture seule.

**Comptabilité file**

Niveau des informations de comptabilité pour les files d'attente. Cet attribut est en lecture seule.

**nom-gestionnaire-files d'attente**

Nom du gestionnaire de files d'attente (éventuellement éloigné) où se trouve la file d'attente.

Ne confondez pas le gestionnaire de files d'attente nommé ici avec la **référence de connexion**

ImqObject , qui fait référence au gestionnaire de files d'attente (local) fournissant une connexion. La valeur initiale est null.

**Ctrl file d'attente**

Niveau de collecte des données de surveillance pour la file d'attente. Cet attribut est en lecture seule.

**statistiques de file d'attente**

Niveau des données statistiques de la file d'attente. Cet attribut est en lecture seule.

**type de file d'attente**

Type de file d'attente. Cet attribut est en lecture seule.

**Nom du gestionnaire de files d'attente éloignées**

Nom du gestionnaire de files d'attente éloignées. Cet attribut est en lecture seule.

**Nom de la file d'attente distante**

Nom de la file d'attente éloignée tel qu'il est connu sur le gestionnaire de files d'attente éloignées. Cet attribut est en lecture seule.

**Nom de gestionnaire de files d'attente résolu**

Nom du gestionnaire de files d'attente résolu. Cet attribut est en lecture seule.

**Nom de file d'attente résolu**

Nom de file d'attente résolu. Cet attribut est en lecture seule.

**intervalle de rétention**

Intervalle de conservation de la file d'attente. Cet attribut est en lecture seule.

**portée**

Portée de la définition de file d'attente. Cet attribut est en lecture seule.

**intervalle de maintenance**

Intervalle de maintenance. Cet attribut est en lecture seule.

**événement d'intervalle de maintenance**

Attribut de contrôle des événements d'intervalle de service. Cet attribut est en lecture seule.

**Partage de file d'attente**

Indique si la file d'attente peut être partagée. Cet attribut est en lecture seule.

**classe d'archivage**

Classe de stockage. Cet attribut est en lecture seule.

**Nom de la file d'attente de transmission**

Nom de la file d'attente de transmission. Cet attribut est en lecture seule.

**Contrôle du déclenchement**

Contrôle du déclencheur. La valeur initiale dépend de la définition de la file d'attente. Cet attribut est valide uniquement pour une file d'attente locale.

**données de déclenchement**

Données de déclenchement. La valeur initiale dépend de la définition de la file d'attente. Cet attribut est valide uniquement pour une file d'attente locale.

**Longueur de déclenchement**

Longueur de déclenchement. La valeur initiale dépend de la définition de la file d'attente. Cet attribut est valide uniquement pour une file d'attente locale.

### **Priorité msg pour déclench.**

Priorité de message de seuil pour les déclencheurs. La valeur initiale dépend de la définition de la file d'attente. Cet attribut est valide uniquement pour une file d'attente locale.

### **type de déclencheur**

Type de déclencheur. La valeur initiale dépend de la définition de la file d'attente. Cet attribut est valide uniquement pour une file d'attente locale.

### **utilisation**

Utilisation. Cet attribut est en lecture seule.

## **Constructeurs**

### **ImqQueue();**

Constructeur par défaut.

### **ImqQueue( const ImqQueue & *file d'attente* );**

Constructeur de copie. L' **état d'ouverture** ImqObject est FALSE.

### **ImqQueue( const char \* *nom* );**

Définit le **nom** ImqObject .

## **Méthodes d'objet (public)**

### **void operator = ( const ImqQueue & *queue* );**

Effectue une fermeture si nécessaire, puis copie les données d'instance à partir de la *file d'attente*. L' **état d'ouverture** ImqObject est FALSE.

### **ImqBoolean backoutRequeueNom ( ImqString & *nom* );**

Fournit une copie du **nom de remise en file d'attente d'annulation**. Elle renvoie TRUE en cas de succès.

### **ImqString backoutRequeueNom ();**

Renvoie le **nom de remise en file d'attente d'annulation** sans aucune indication d'erreurs possibles.

### **ImqBoolean backoutThreshold ( MQLONG & *seuil* );**

Fournit une copie du **seuil d'annulation**. Elle renvoie TRUE en cas de succès.

### **MQLONG backoutThreshold ();**

Renvoie la valeur de **seuil d'annulation** sans indication d'erreurs possibles.

### **ImqBoolean baseQueueNom ( ImqString & *nom* );**

Fournit une copie du **nom de la file d'attente de base**. Elle renvoie TRUE en cas de succès.

### **ImqString baseQueueNom ();**

Renvoie le **nom de la file d'attente de base** sans indication des erreurs possibles.

### **ImqBoolean clusterName( ImqString & *nom* );**

Fournit une copie du **nom de cluster**. Elle renvoie TRUE en cas de succès.

### **ImqString clusterName();**

Renvoie le **nom de cluster** sans aucune indication d'erreurs possibles.

### **ImqBoolean clusterNamelistNom ( ImqString & *nom* );**

Fournit une copie du **nom de liste de noms de cluster**. Elle renvoie TRUE en cas de succès.

### **ImqString clusterNamelistNom ();**

Renvoie le **nom de liste de noms de cluster** sans aucune indication d'erreurs.

### **ImqBoolean clusterWorkLoadPriority ( MQLONG & *priorité* );**

Fournit une copie de la valeur de priorité de charge de travail du cluster. Elle renvoie TRUE en cas de succès.

### **MQLONG clusterWorkLoadPriority ();**

Renvoie la valeur de priorité de la charge de travail du cluster sans aucune indication d'erreurs possibles.

**ImqBoolean clusterWorkLoadRank ( MQLONG & rank ) ;**

Fournit une copie de la valeur de rang de charge de travail du cluster. Elle renvoie TRUE en cas de succès.

**MQLONG clusterWorkLoadRank () ;**

Renvoie la valeur de rang de charge de travail du cluster sans aucune indication d'erreurs possibles.

**ImqBoolean clusterWorkLoadUseQ ( MQLONG & useq ) ;**

Fournit une copie de la valeur de la file d'attente d'utilisation de la charge de travail du cluster. Elle renvoie TRUE en cas de succès.

**MQLONG clusterWorkLoadUseQ () ;**

Renvoie la valeur de la file d'attente d'utilisation de la charge de travail du cluster sans aucune indication d'erreurs possibles.

**ImqBoolean creationDate ( ImqString & date ) ;**

Fournit une copie de la **date de création**. Elle renvoie TRUE en cas de succès.

**ImqString creationDate () ;**

Renvoie la **date de création** sans aucune indication d'erreurs possibles.

**ImqBoolean creationTime ( ImqString & heure ) ;**

Fournit une copie de l' **heure de création**. Elle renvoie TRUE en cas de succès.

**ImqString creationTime () ;**

Renvoie l' **heure de création** sans aucune indication des erreurs possibles.

**ImqBoolean currentDepth ( MQLONG & profondeur ) ;**

Fournit une copie de la **profondeur actuelle**. Elle renvoie TRUE en cas de succès.

**MQLONG currentDepth () ;**

Renvoie la **profondeur en cours** sans aucune indication d'erreurs possibles.

**ImqBoolean defaultInputOpenOption ( MQLONG & option ) ;**

Fournit une copie de l' **option d'ouverture d'entrée par défaut**. Elle renvoie TRUE en cas de succès.

**MQLONG defaultInputOpenOption () ;**

Renvoie l' **option d'ouverture d'entrée par défaut** sans aucune indication d'erreurs possibles.

**ImqBoolean defaultPersistence ( MQLONG & persistance ) ;**

Fournit une copie de la **persistance par défaut**. Elle renvoie TRUE en cas de succès.

**MQLONG defaultPersistence () ;**

Renvoie la **persistance par défaut** sans indication des erreurs possibles.

**ImqBoolean defaultPriority ( MQLONG & priorité ) ;**

Fournit une copie de la **priorité par défaut**. Elle renvoie TRUE en cas de succès.

**MQLONG defaultPriority () ;**

Renvoie la **priorité par défaut** sans aucune indication d'erreurs possibles.

**ImqBoolean defaultBind ( MQLONG & liaison ) ;**

Fournit une copie de la **liaison par défaut**. Elle renvoie TRUE en cas de succès.

**MQLONG defaultBind () ;**

Renvoie la **liaison par défaut** sans aucune indication d'erreurs possibles.

**ImqBoolean definitionType ( MQLONG & type ) ;**

Fournit une copie du **type de définition**. Elle renvoie TRUE en cas de succès.

**MQLONG definitionType () ;**

Renvoie le **type de définition** sans aucune indication d'erreurs possibles.

**ImqBoolean depthHighEvénement ( MQLONG & événement ) ;**

Fournit une copie de l'état d'activation de l' **événement de profondeur élevée**. Elle renvoie TRUE en cas de succès.

**MQLONG depthHighEvénement () ;**

Renvoie l'état d'activation de l' **événement de profondeur élevée** sans aucune indication d'erreurs possibles.

**ImqBoolean depthHighLimit ( MQLONG & limite ) ;**

Fournit une copie de la **limite supérieure de profondeur**. Elle renvoie TRUE en cas de succès.

**MQLONG depthHighLimite () ;**

Renvoie la valeur de la **limite supérieure de profondeur** sans aucune indication d'erreurs possibles.

**ImqBoolean depthLowEvénement ( MQLONG & événement ) ;**

Fournit une copie de l'état d'activation de l' **événement de profondeur faible**. Elle renvoie TRUE en cas de succès.

**MQLONG depthLowEvénement () ;**

Renvoie l'état d'activation de l' **événement de profondeur faible** sans aucune indication d'erreurs possibles.

**ImqBoolean depthLowLimite ( MQLONG & limite ) ;**

Fournit une copie de la **limite inférieure de profondeur**. Elle renvoie TRUE en cas de succès.

**MQLONG depthLowLimite () ;**

Renvoie la valeur de la **limite inférieure de profondeur** sans aucune indication d'erreurs possibles.

**ImqBoolean depthMaximumEvénement ( MQLONG & événement ) ;**

Fournit une copie de l'état d'activation de l' **événement maximal de profondeur**. Elle renvoie TRUE en cas de succès.

**MQLONG depthMaximumEvénement () ;**

Renvoie l'état d'activation de l' **événement maximal de profondeur** sans aucune indication d'erreurs possibles.

**ImqDistributionListe \* distributionListRéférence () const ;**

Renvoie la **référence de liste de distribution**.

**void setDistributionListReference ( ImqDistributionList & list ) ;**

Définit la **référence de liste de distribution**.

**void setDistributionListReference ( ImqDistributionList \* list = 0 ) ;**

Définit la **référence de liste de distribution**.

**ImqBoolean distributionLists ( MQLONG & prise en charge ) ;**

Fournit une copie de la valeur des **listes de distribution** . Elle renvoie TRUE en cas de succès.

**MQLONG distributionLists () ;**

Renvoie la valeur de **listes de distribution** sans aucune indication d'erreurs possibles.

**ImqBoolean setDistributionLists ( const MQLONG support ) ;**

Définit la valeur des **listes de distribution** . Elle renvoie TRUE en cas de succès.

**ImqString dynamicQueueNom () const ;**

Renvoie une copie du **nom de file d'attente dynamique**.

**ImqBoolean setDynamicQueueName ( const char \* nom ) ;**

Définit le **nom de la file d'attente dynamique**. Le **nom de file d'attente dynamique** ne peut être défini que si le **statut d'ouverture** ImqObject est FALSE. Elle renvoie TRUE en cas de succès.

**ImqBoolean get ( ImqMessage & msg, ImqGetMessageOptions & options ) ;**

Extrait un message de la file d'attente à l'aide des *option*s spécifiées. Appelle la méthode ImqObject **openFor** si nécessaire pour s'assurer que les **options d'ouverture** ImqObject incluent l'une des valeurs MQOO\_INPUT\_ \* ou MQOO\_BROWSE, en fonction des *options*. Si l'objet *msg* possède une **mémoire tampon automatique** ImqCache , la taille de la mémoire tampon est adaptée à tous les messages extraits. La méthode **clearMessage** est appelée sur l'objet *msg* avant l'extraction.

Cette méthode renvoie TRUE en cas de succès.

**Remarque :** Le résultat de l'appel de méthode est FALSE si le code anomalie ImqObject est MQRC\_TRUNCATED\_MSG\_FAILED, même si ce **code anomalie** est classé comme un avertissement. Si un message tronqué est accepté, la **longueur de message** ImqCache reflète la longueur tronquée. Dans les deux cas, la **longueur totale du message** ImqMessage indique le nombre d'octets disponibles.

**ImqBoolean get ( ImqMessage & msg ) ;**

Comme pour la méthode précédente, sauf que les options d'obtention de message par défaut sont utilisées.

**ImqBoolean get ( ImqMessage & msg, ImqGetMessageOptions & options, const size\_t buffer-size ) ;**

Comme pour les deux méthodes précédentes, sauf qu'une *taille de mémoire tampon* de substitution est indiquée. Si l'objet *msg* emploie une **mémoire tampon automatique** *ImqCache* , la méthode **resizeBuffer** est appelée sur l'objet *msg* avant l'extraction des messages, et la mémoire tampon n'augmente pas davantage pour contenir les messages plus volumineux.

**ImqBoolean get ( ImqMessage & msg, const size\_t taille-tampon ) ;**

Comme pour la méthode précédente, sauf que les options d'obtention de message par défaut sont utilisées.

**ImqBoolean hardenGetBackout ( MQLONG & harden ) ;**

Fournit une copie de la valeur **harden get backout** . Elle renvoie TRUE en cas de succès.

**MQLONG hardenGetAnnulation () ;**

Renvoie la valeur **harden get backout** sans aucune indication d'erreurs possibles.

**ImqBoolean indexType(MQLONG & type ) ;**

Fournit une copie du **type d'index**. Elle renvoie TRUE en cas de succès.

**MQLONG indexType() ;**

Renvoie le **type d'index** sans aucune indication des erreurs possibles.

**ImqBoolean inhibitGet ( MQLONG & inhibition ) ;**

Fournit une copie de la valeur **inhibition de l'extraction** . Elle renvoie TRUE en cas de succès.

**MQLONG inhibitGet () ;**

Renvoie la valeur **inhibant l'obtention** sans aucune indication d'erreurs possibles.

**ImqBoolean setInhibitGet ( const MQLONG inhibition ) ;**

Définit la valeur **inhibition de l'extraction** . Elle renvoie TRUE en cas de succès.

**ImqBoolean inhibitPut ( MQLONG & inhibition ) ;**

Fournit une copie de la valeur **inhibition de l'insertion** . Elle renvoie TRUE en cas de succès.

**MQLONG inhibitPut () ;**

Renvoie la valeur **inhibant l'insertion** sans aucune indication d'erreurs possibles.

**ImqBoolean setInhibitPut ( const MQLONG inhibition ) ;**

Définit la valeur **inhibant l'insertion** . Elle renvoie TRUE en cas de succès.

**ImqBoolean initiationQueueNom ( ImqString & nom ) ;**

Fournit une copie du **nom de la file d'attente d'initialisation**. Elle renvoie TRUE en cas de succès.

**ImqString initiationQueueNom () ;**

Renvoie le **nom de la file d'attente d'initialisation** sans aucune indication d'erreurs possibles.

**ImqBoolean maximumDepth ( MQLONG & profondeur ) ;**

Fournit une copie de la **profondeur maximale**. Elle renvoie TRUE en cas de succès.

**MQLONG maximumDepth () ;**

Renvoie la **profondeur maximale** sans indication des erreurs possibles.

**ImqBoolean maximumMessageLongueur ( MQLONG & longueur ) ;**

Fournit une copie de la **longueur maximale de message**. Elle renvoie TRUE en cas de succès.

**MQLONG maximumMessageLongueur () ;**

Renvoie la **longueur maximale de message** sans aucune indication d'erreurs possibles.

**ImqBoolean messageDeliverySéquence ( MQLONG & séquence ) ;**

Fournit une copie de la **séquence de distribution des messages**. Elle renvoie TRUE en cas de succès.

**MQLONG messageDeliverySéquence () ;**

Renvoie la valeur de la **séquence de distribution des messages** sans aucune indication d'erreurs possibles.

**ImqQueue \* nextDistributedFile d'attente () const ;**

Renvoie la **file d'attente distribuée suivante**.

**ImqBoolean nonPersistentMessageClass (MQLONG & monq) ;**

Fournit une copie de la valeur de classe de message non persistant. Elle renvoie TRUE en cas de succès.

**MQLONG nonPersistentMessageClass () ;**

Renvoie la valeur de classe de message non persistant sans aucune indication d'erreurs possibles.

**ImqBoolean openInputNombre ( MQLONG & nombre ) ;**

Fournit une copie du **nombre d'entrées ouvertes**. Elle renvoie TRUE en cas de succès.

**MQLONG openInputNombre () ;**

Renvoie le **nombre d'entrées ouvertes** sans aucune indication d'erreurs possibles.

**ImqBoolean openOutputNombre ( MQLONG & nombre ) ;**

Fournit une copie du **nombre de sorties ouvertes**. Elle renvoie TRUE en cas de succès.

**MQLONG openOutputNombre () ;**

Renvoie le **nombre de sorties ouvertes** sans aucune indication d'erreurs possibles.

**ImqQueue \* previousDistributedprécédente distribuée () const ;**

Renvoie la **file d'attente répartie précédente**.

**ImqBoolean processName ( ImqString & nom ) ;**

Fournit une copie du **nom de processus**. Elle renvoie TRUE en cas de succès.

**ImqString processName () ;**

Renvoie le **nom de processus** sans aucune indication des erreurs possibles.

**ImqBoolean put ( ImqMessage & msg ) ;**

Place un message dans la file d'attente à l'aide des options de message d'insertion par défaut. Utilise la méthode ImqObject **openFor** si nécessaire pour s'assurer que les **options d'ouverture** ImqObject incluent MQOO\_OUTPUT.

Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean put ( ImqMessage & msg, ImqPutMessageOptions & pmo ) ;**

Place un message dans la file d'attente, en utilisant le *pmo* spécifié. Utilise la méthode ImqObject **openFor** si nécessaire pour s'assurer que les **options d'ouverture** ImqObject incluent MQOO\_OUTPUT et (si les options *pmo* incluent l'une des valeurs MQPMO\_PASS\_IDENTITY\_CONTEXT, MQPMO\_PASS\_ALL\_CONTEXT, MQPMO\_SET\_IDENTITY\_CONTEXT ou MQPMO\_SET\_ALL\_CONTEXT) correspondantes MQOO\_\*\_CONTEXT.

Cette méthode renvoie TRUE en cas de succès.

**Remarque :** Si *pmo* inclut une **référence de contexte**, l'objet référencé est ouvert, si nécessaire, pour fournir un contexte.

**ImqBoolean queueAccounting (MQLONG & acctq) ;**

Fournit une copie de la valeur de comptabilité de la file d'attente. Elle renvoie TRUE en cas de succès.

**MQLONG queueAccounting () ;**

Renvoie la valeur de comptabilité de la file d'attente sans indication d'erreurs possibles.

**ImqString queueManagerNom () const ;**

Renvoie le **nom du gestionnaire de files d'attente**.

**ImqBoolean setQueueManagerName ( const char \* name ) ;**

Définit le **nom du gestionnaire de files d'attente**. Le **nom du gestionnaire de files d'attente** ne peut être défini que si le **statut d'ouverture** ImqObject est FALSE. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean queueMonitoring (MQLONG & monq) ;**

Fournit une copie de la valeur de surveillance de file d'attente. Elle renvoie TRUE en cas de succès.

**MQLONG queueMonitoring () ;**

Renvoie la valeur de surveillance de file d'attente sans aucune indication d'erreurs possibles.

**ImqBoolean queueStatistics (MQLONG & statq) ;**

Fournit une copie de la valeur des statistiques de file d'attente. Elle renvoie TRUE en cas de succès.

**MQLONG queueStatistics () ;**

Renvoie la valeur des statistiques de file d'attente sans indication d'erreurs possibles.

**ImqBoolean queueType ( MQLONG & type ) ;**

Fournit une copie de la valeur du **type de file d'attente** . Elle renvoie TRUE en cas de succès.

**MQLONG queueType () ;**

Renvoie le **type de file d'attente** sans aucune indication d'erreurs possibles.

**ImqBoolean remoteQueueManagerName ( ImqString & name ) ;**

Fournit une copie du **nom du gestionnaire de files d'attente éloignées**. Elle renvoie TRUE en cas de succès.

**ImqString remoteQueueManagerName () ;**

Renvoie le **nom du gestionnaire de files d'attente éloignées** sans indiquer d'erreurs possibles.

**ImqBoolean remoteQueueNom ( ImqString & nom ) ;**

Fournit une copie du **nom de la file d'attente éloignée**. Elle renvoie TRUE en cas de succès.

**ImqString remoteQueueNom () ;**

Renvoie le **nom de la file d'attente éloignée** sans indiquer d'erreurs possibles.

**ImqBoolean resolvedQueueManagerName( ImqString & name ) ;**

Fournit une copie du **nom du gestionnaire de files d'attente résolu**. Elle renvoie TRUE en cas de succès.

**Remarque :** Cette méthode échoue sauf si MQOO\_RESOLVE\_NAMES fait partie des **options d'ouverture** ImqObject .

**ImqString resolvedQueueManagerName() ;**

Renvoie le **nom du gestionnaire de files d'attente résolu**, sans aucune indication d'erreurs possibles.

**ImqBoolean resolvedQueueNom ( ImqString & nom ) ;**

Fournit une copie du **nom de file d'attente résolue**. Elle renvoie TRUE en cas de succès.

**Remarque :** Cette méthode échoue sauf si MQOO\_RESOLVE\_NAMES fait partie des **options d'ouverture** ImqObject .

**ImqString resolvedQueueNom () ;**

Renvoie le **nom de la file d'attente résolue**, sans aucune indication d'erreurs possibles.

**ImqBoolean retentionInterval ( MQLONG & intervalle ) ;**

Fournit une copie de l' **intervalle de conservation**. Elle renvoie TRUE en cas de succès.

**MQLONG retentionInterval () ;**

Renvoie l' **intervalle de conservation** sans aucune indication d'erreurs possibles.

**ImqBoolean scope ( MQLONG & portée ) ;**

Fournit une copie de la **portée**. Elle renvoie TRUE en cas de succès.

**MQLONG portée () ;**

Renvoie la **portée** sans indication des erreurs possibles.

**ImqBoolean serviceInterval ( MQLONG & intervalle ) ;**

Fournit une copie de l' **intervalle de service**. Elle renvoie TRUE en cas de succès.

**MQLONG serviceInterval () ;**

Renvoie l' **intervalle de service** sans aucune indication d'erreurs possibles.

**ImqBoolean serviceIntervalÉvénement ( MQLONG & événement ) ;**

Fournit une copie de l'état d'activation de l' **événement d'intervalle de service**. Elle renvoie TRUE en cas de succès.

**MQLONG serviceIntervalÉvénement () ;**

Renvoie l'état d'activation de l' **événement d'intervalle de service** sans aucune indication d'erreurs possibles.

**ImqBoolean partageabilité ( MQLONG & partageabilité ) ;**

Fournit une copie de la valeur **shareability** . Elle renvoie TRUE en cas de succès.

**MQLONG partageabilité () ;**

Renvoie la valeur **shareability** sans aucune indication des erreurs possibles.



**ImqBoolean storageClass( ImqString & classe ) ;**

Fournit une copie de la **classe de stockage**. Elle renvoie TRUE en cas de succès.

**ImqString storageClass( ) ;**

Renvoie la **classe de stockage** sans aucune indication d'erreurs possibles.

**ImqBoolean transmissionQueueNom ( ImqString & nom ) ;**

Fournit une copie du **nom de la file d'attente de transmission**. Elle renvoie TRUE en cas de succès.

**ImqString transmissionQueueNom ( ) ;**

Renvoie le **nom de la file d'attente de transmission** sans aucune indication d'erreurs possibles.

**ImqBoolean triggerControl ( MQLONG & contrôle ) ;**

Fournit une copie de la valeur **trigger control** . Elle renvoie TRUE en cas de succès.

**MQLONG triggerControl ( ) ;**

Renvoie la valeur **trigger control** sans aucune indication d'erreurs possibles.

**ImqBoolean setTriggerControl ( const MQLONG contrôle ) ;**

Définit la valeur du **contrôle de déclencheur** . Elle renvoie TRUE en cas de succès.

**ImqBoolean triggerData ( ImqString & données ) ;**

Fournit une copie des **données de déclenchement**. Elle renvoie TRUE en cas de succès.

**ImqString triggerData ( ) ;**

Renvoie une copie des **données de déclencheur** sans indication d'erreurs possibles.

**ImqBoolean setTriggerData ( const char \* data ) ;**

Définit les **données de déclenchement**. Elle renvoie TRUE en cas de succès.

**ImqBoolean triggerDepth ( MQLONG & profondeur ) ;**

Fournit une copie de la **profondeur du déclencheur**. Elle renvoie TRUE en cas de succès.

**MQLONG triggerDepth ( ) ;**

Renvoie la **profondeur du déclencheur** sans aucune indication d'erreurs possibles.

**ImqBoolean setTriggerDepth ( const MQLONG profondeur ) ;**

Définit la **profondeur du déclencheur**. Elle renvoie TRUE en cas de succès.

**ImqBoolean triggerMessagePriorité ( MQLONG & priorité ) ;**

Fournit une copie de la **priorité du message de déclenchement**. Elle renvoie TRUE en cas de succès.

**MQLONG triggerMessagePriorité ( ) ;**

Renvoie la **priorité de message de déclenchement** sans aucune indication d'erreurs possibles.

**ImqBoolean setTriggerMessagePriority ( const MQLONG priorité ) ;**

Définit la **priorité des messages de déclenchement**. Elle renvoie TRUE en cas de succès.

**ImqBoolean triggerType ( MQLONG & type ) ;**

Fournit une copie du **type de déclencheur**. Elle renvoie TRUE en cas de succès.

**MQLONG triggerType ( ) ;**

Renvoie le **type de déclencheur** sans aucune indication d'erreurs possibles.

**ImqBoolean setTriggerType ( const MQLONG type ) ;**

Définit le **type de déclencheur**. Elle renvoie TRUE en cas de succès.

**ImqBoolean usage ( MQLONG & usage ) ;**

Fournit une copie de la valeur **usage** . Elle renvoie TRUE en cas de succès.

**MQLONG syntaxe ( ) ;**

Renvoie la valeur **usage** sans aucune indication d'erreurs possibles.

## Méthodes d'objet (protégées)

**void setNextDistributedQueue ( ImqQueue \* queue = 0 ) ;**

Définit la **file d'attente répartie suivante**.

**Attention:** utilisez cette fonction uniquement si vous êtes sûr qu'elle n'interrompt pas la liste des files d'attente réparties.

**void setPreviousDistributedQueue ( ImqQueue \* queue = 0 ) ;**

Définit la **file d'attente distribuée précédente**.

**Attention:** utilisez cette fonction uniquement si vous êtes sûr qu'elle n'interrompt pas la liste des files d'attente réparties.

### Codes raison

- MQRC\_ATTRIBUT\_LOCKED
- MQRC\_CONTEXT\_OBJET\_NON\_VALIDE
- MQRC\_CONTEXT\_OPEN\_ERROR
- MQRC\_CURSOR\_NOT\_VALID
- MQRC\_NO\_BUFFER
- MQRC\_REOPEN\_EXCL\_INPUT\_ERROR
- MQRC\_REOPEN\_INQUIRE\_ERROR
- MQRC\_REOPEN\_TEMPORARY\_Q\_ERROR
- (codes anomalie de MQGET)
- (codes anomalie de MQPUT)

## Classe C++ du gestionnaire ImqQueue

Cette classe encapsule un gestionnaire de files d'attente (objet IBM MQ de type MQOT\_Q\_MGR).

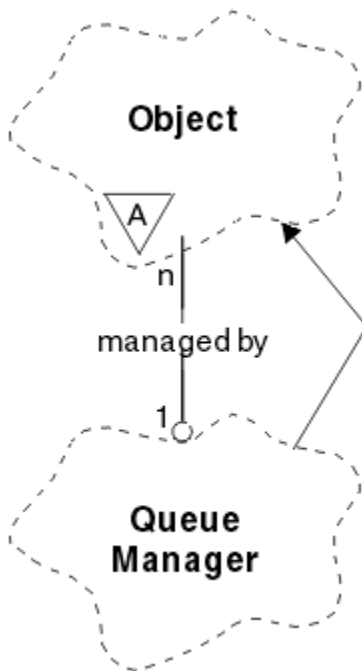


Figure 33. Classe de gestionnaire ImqQueue

Cette classe est liée aux appels MQI répertoriés dans «Références croisées du gestionnaire ImqQueue», à la page 1882. Toutes les méthodes répertoriées ne sont pas applicables à toutes les plateformes ; voir [ALTER QMGR](#) pour plus de détails.

- [«Attributs de classe»](#), à la page 1955
- [«Attributs d'objet»](#), à la page 1955
- [«Constructeurs»](#), à la page 1961
- [«Destructeurs»](#), à la page 1961

- «Méthodes de classe (public)», à la page 1961
- «Méthodes d'objet (public)», à la page 1961
- «Méthodes d'objet (protégées)», à la page 1970
- «Données d'objet (protégées)», à la page 1970
- «Codes raison», à la page 1970

## Attributs de classe

### comportement

Contrôle le comportement de la connexion et de la déconnexion implicites.

#### **IMQ\_EXPL\_DISC\_BACKOUT (0L)**

Un appel explicite à la méthode de déconnexion implique une annulation. Cet attribut et IMQ\_EXPL\_DISC\_COMMIT s'excluent mutuellement.

#### **IMQ\_EXPL\_DISC\_COMMIT (1L)**

Un appel explicite à la méthode de déconnexion implique une validation (valeur par défaut). Cet attribut et IMQ\_EXPL\_DISC\_BACKOUT s'excluent mutuellement.

#### **IMQ\_IMPL\_CONN (2L)**

La connexion implicite est autorisée (valeur par défaut).

#### **IMQ\_IMPL\_DISC\_BACKOUT (0L)**

Un appel implicite à la méthode de déconnexion, qui peut se produire lors de la destruction de l'objet, implique une annulation. Cet attribut est mutuellement exclusif avec IMQ\_IMPL\_DISC\_COMMIT.

#### **IMQ\_IMPL\_DISC\_COMMIT (4L)**

Un appel implicite à la méthode de déconnexion, qui peut se produire lors de la destruction d'objet, implique la validation (valeur par défaut). Cet attribut et IMQ\_IMPL\_DISC\_BACKOUT s'excluent mutuellement.

Dans IBM MQ V7.0 et les versions ultérieures, les applications C++ qui utilisent une connexion implicite doivent spécifier IMQ\_IMPL\_CONN avec les autres options fournies dans la méthode `setBehavior()` sur un objet de classe `ImqQueueManager`. Si votre application n'utilise pas la méthode `setBehavior()` pour définir explicitement les options de comportement, par exemple,

```
ImqQueueManager_object.setBehavior(IMQ_IMPL_DISC_COMMIT)
```

Cette modification ne vous affecte pas car IMQ\_IMPL\_CONN est activé par défaut.

Si votre application définit explicitement les options de comportement, par exemple,

```
ImqQueueManager_object.setBehavior(IMQ_IMPL_DISC_COMMIT)
```

vous devez inclure IMQ\_IMPL\_CONN dans la méthode `setBehavior()` comme suit pour permettre à votre application d'établir une connexion implicite:

```
ImqQueueManager_object.setBehavior(IMQ_IMPL_CONN | IMQ_IMPL_DISC_COMMIT)
```

## Attributs d'objet

### remplacement des connexions de comptabilité

Permet aux applications de remplacer les valeurs values.This est en lecture seule.

### intervalle de comptabilité

Durée avant l'écriture des enregistrements comptables intermédiaires (en secondes). Cet attribut est en lecture seule.

### enregistrement de l'activité

Ctrl la génération des rapports d'activité. Cet attribut est en lecture seule.

**Adoption de la vérification du nouveau MCA**

Les éléments vérifiés pour déterminer si un agent MCA doit être adopté lorsqu'un nouveau canal entrant portant le même nom qu'un agent MCA déjà actif est détecté. Cet attribut est en lecture seule.

**Adoption du nouveau type de MCA**

Indique si une instance orpheline d'un agent MCA d'un type de canal particulier doit être redémarrée automatiquement lorsqu'une nouvelle demande de canal entrant correspondant aux nouveaux paramètres de vérification mca est détectée. Cet attribut est en lecture seule.

**type d'authentification**

Indique le type d'authentification en cours d'exécution.

**événement d'autorité**

Contrôle les événements de droits d'accès. Cet attribut est en lecture seule.

**Options de début**

Options qui s'appliquent à la méthode begin. La valeur initiale est MQBO\_NONE.

**événement de pont**

Indique si les événements de pont IMS sont générés. Cet attribut est en lecture seule.

**Définition automatique de canal**

Valeur de définition automatique de canal. Cet attribut est en lecture seule.

**événement de définition automatique de canal**

Valeur d'événement de définition automatique de canal. Cet attribut est en lecture seule.

**Exit de définition automatique de canal**

Nom de l'exit de définition automatique de canal. Cet attribut est en lecture seule.

**événement Canal**

Indique si des événements de canal sont générés. Cet attribut est en lecture seule.

**Adaptateurs d'initiateur de canal**

Nombre de sous-tâches d'adaptateur à utiliser pour le traitement des appels IBM MQ . Cet attribut est en lecture seule.

**Ctrl init. canal**

Indique si l'initialisateur de canal doit être démarré automatiquement au démarrage du gestionnaire de files d'attente. Cet attribut est en lecture seule.

**Répartiteurs d'initiateur de canal**

Nombre de répartiteurs à utiliser pour l'initiateur de canal. Cet attribut est en lecture seule.

**démarrage automatique de la trace de l'initiateur de canal**

Indique si la trace de l'initiateur de canal doit démarrer automatiquement ou non. Cet attribut est en lecture seule.

**Taille de la table de trace de l'initiateur de canal**

Taille de l'espace de données de trace de l'initiateur de canal (en Mo). Cet attribut est en lecture seule.

**Contrôle des canaux**

Ctrl la collecte des données de surveillance en ligne pour les canaux. Cet attribut est en lecture seule.

**référence de canal**

Référence à une définition de canal à utiliser lors de la connexion client. Lorsqu'il est connecté, cet attribut peut être défini sur null, mais ne peut pas être remplacé par une autre valeur. La valeur initiale est null.

**statistiques de canal**

Ctrl la collecte des données de statistiques pour les canaux. Cet attribut est en lecture seule.

**jeu de caractères**

ID de jeu de caractères codés (CCSID). Cet attribut est en lecture seule.

**Ctrl émetteur cluster**

Contrôle la collecte des données de surveillance en ligne pour les canaux émetteurs de cluster définis automatiquement. Cet attribut est en lecture seule.

**Stats émetteur cluster**

Contrôle la collecte des données statistiques pour les canaux émetteurs de cluster définis automatiquement. Cet attribut est en lecture seule.

**Données de charge de travail de cluster**

Données d'exit de charge de travail de cluster. Cet attribut est en lecture seule.

**Exit de charge de travail du cluster**

Nom de l'exit de charge de travail de cluster. Cet attribut est en lecture seule.

**Longueur de charge de travail de cluster**

Longueur de la charge de travail du cluster. Cet attribut est en lecture seule.

**mru de charge de travail de cluster**

La charge de travail du cluster a utilisé la valeur de canaux la plus récente. Cet attribut est en lecture seule.

**File d'attente d'utilisation de charge de travail de cluster**

Valeur de file d'attente d'utilisation de charge de travail de cluster Cet attribut est en lecture seule.

**événement de commande**

Indique si des événements de commande sont générés. Cet attribut est en lecture seule.

**Nom de la file d'attente d'entrée des commandes**

Nom de la file d'attente d'entrée des commandes système. Cet attribut est en lecture seule.

**niveau de commande**

Niveau de commande pris en charge par le gestionnaire de files d'attente. Cet attribut est en lecture seule.

**Contrôle du serveur de commandes**

Indique si le serveur de commandes doit être démarré automatiquement lorsque le gestionnaire de files d'attente est démarré. Cet attribut est en lecture seule.

**Options de connexion**

Options qui s'appliquent à la méthode de connexion. La valeur initiale est MQCNO\_NONE. Les valeurs supplémentaires suivantes peuvent être possibles en fonction de la plateforme:

- MQCNO\_STANDARD\_BINDING
- MQCNO\_FASTPATH\_BINDING
- MQCNO\_HANDLE\_SHARE\_NONE
- MQCNO\_HANDLE\_SHARE\_BLOCK
- MQCNO\_HANDLE\_SHARE\_NO\_BLOCK
- MQCNO\_SERIALIZE\_CONN\_TAG\_Q\_MGR
- MQCNO\_SERIALIZE\_CONN\_TAG\_QSG
- MQCNO\_RESTRICT\_CONN\_TAG\_Q\_MGR
- MQCNO\_RESTRICT\_CONN\_TAG\_QSG

**ID de connexion**

Identificateur unique qui permet à MQ d'identifier de manière fiable une application.

**état de connexion**

TRUE en cas de connexion au gestionnaire de files d'attente. Cet attribut est en lecture seule.

**balise de connexion**

Balise à associer à une connexion. Cet attribut ne peut être défini que s'il n'est pas connecté. La valeur initiale est null.

**matériel de cryptographie**

Détails de configuration du matériel de cryptographie. Pour les connexions client MQ MQI.

**Nom de la file d'attente de rebut**

Nom de la file d'attente de rebut. Cet attribut est en lecture seule.

**nom de file de transmission par défaut**

Nom de la file d'attente de transmission par défaut. Cet attribut est en lecture seule.

**listes de distribution**

Capacité du gestionnaire de files d'attente à prendre en charge les listes de distribution.

**groupe dns**

Nom du groupe que le programme d'écoute TCP qui gère les transmissions entrantes pour le groupe de partage de files d'attente doit joindre lors de l'utilisation de la prise en charge des services de nom de domaine dynamique de Workload Manager. Cet attribut est en lecture seule.

**dns wlm**

Indique si le programme d'écoute TCP qui gère les transmissions entrantes pour le groupe de partage de files d'attente doit s'enregistrer auprès de Workload Manager for Dynamic Domain Name Services. Cet attribut est en lecture seule.

**premier enregistrement d'authentification**

Le premier d'un ou plusieurs objets de la classe ImqAuthenticationRecord, sans ordre particulier, dans lequel la référence de connexion d'enregistrement ImqAuthenticationadresse cet objet. Pour les connexions client MQ MQI.

**premier objet géré**

Le premier d'un ou de plusieurs objets de la classe ImqObject, sans ordre particulier, dans lequel la référence de connexion ImqObject adresse cet objet. La valeur initiale est zéro.

**événement d'interdiction**

Contrôle les événements d'interdiction. Cet attribut est en lecture seule.

**Version adresse IP**

Protocole IP (IPv4 ou IPv6) à utiliser pour une connexion de canal. Cet attribut est en lecture seule.

**référentiel de clés**

Emplacement du fichier de base de données de clés dans lequel les clés et les certificats sont stockés. Pour les connexions IBM MQ MQI client .

**nombre de réinitialisations de clé**

Nombre d'octets non chiffrés envoyés et reçus dans une conversation TLS avant la renégociation de la clé secrète. Cet attribut s'applique uniquement aux connexions client utilisant MQCONN. Voir aussi [ssl key reset count](#).

**Délai du programme d'écoute**

Intervalle (en secondes) entre les tentatives de redémarrage du programme d'écoute par IBM MQ en cas d'échec APPC ou TCP/IP. Cet attribut est en lecture seule.

**événement local**

Contrôle les événements locaux. Cet attribut est en lecture seule.

**événement de consignateur**

Contrôle si les événements du journal de reprise sont générés. Cet attribut est en lecture seule.

**Nom de groupe de LU**

Nom d'unité logique générique que le programme d'écoute d'unité logique 6.2 qui gère les transmissions entrantes pour le groupe de partage de files d'attente doit utiliser. Cet attribut est en lecture seule.

**Nom de LU**

Nom de l'unité logique à utiliser pour les transmissions LU 6.2 sortantes. Cet attribut est en lecture seule.

**Suffixe de bras lu62**

Suffixe de SYS1.PARMLIB APPCPMxx, qui désigne le LUADD pour cet initiateur de canal. Cet attribut est en lecture seule.

**Canaux lu62**

Nombre maximal de canaux pouvant être en cours ou de clients pouvant être connectés, qui utilisent le protocole de transmission LU 6.2 . Cet attribut est en lecture seule.

**nombre maximal de canaux actifs**

Le nombre maximal de canaux pouvant être actifs en même temps. Cet attribut est en lecture seule.

**Nombre maximal de canaux**

Le nombre maximal de canaux pouvant être actifs (notamment les canaux de connexion serveur avec des clients connectés). Cet attribut est en lecture seule.

**nombre maximal de descripteurs**

Nombre maximal de descripteurs. Cet attribut est en lecture seule.

**longueur maximale des messages**

Longueur maximale possible pour tout message d'une file d'attente gérée par ce gestionnaire de files d'attente. Cet attribut est en lecture seule.

**Priorité maximum**

Priorité maximale des messages. Cet attribut est en lecture seule.

**Nb max. messages non validés**

Nombre maximal de messages non validés dans une unité ou un travail. Cet attribut est en lecture seule.

**Comptabilité MQI**

Ctrl la collecte des informations comptables pour les données MQI. Cet attribut est en lecture seule.

**Statistiques MQI**

Ctrl la collecte d'informations de contrôle de statistiques pour le gestionnaire de files d'attente. Cet attribut est en lecture seule.

**port de communications sortantes maximum**

Extrémité supérieure de la plage de numéros de port à utiliser lors de la liaison de canaux sortants. Cet attribut est en lecture seule.

**port de communications sortantes minimum**

Extrémité inférieure de la plage de numéros de port à utiliser lors de la liaison des canaux sortants. Cet attribut est en lecture seule.

**Mot de passe**

mot de passe associé à l'ID utilisateur

**événement Performances**

Contrôle les événements de performances. Cet attribut est en lecture seule.

**plateforme**

Plateforme sur laquelle réside le gestionnaire de files d'attente. Cet attribut est en lecture seule.

**Comptabilité file**

Ctrl la collecte des informations comptables pour les files d'attente. Cet attribut est en lecture seule.

**Ctrl file d'attente**

Ctrl la collecte des données de surveillance en ligne pour les files d'attente. Cet attribut est en lecture seule.

**statistiques de file d'attente**

Ctrl la collecte des données de statistiques pour les files d'attente. Cet attribut est en lecture seule.

**délai d'attente de réception**

Durée approximative pendant laquelle un canal de transmission de messages TCP/IP attend la réception de données, y compris les signaux de présence, de la part de son partenaire, avant de revenir à l'état inactif. Cet attribut est en lecture seule.

**délai d'attente minimal de réception**

Durée minimale pendant laquelle un canal TCP/IP attend de recevoir des données, y compris des signaux de présence, de la part de son partenaire, avant de revenir à l'état inactif. Cet attribut est en lecture seule.

**Type de délai de réception**

Qualificateur appliqué pour le délai d'attente de réception. Cet attribut est en lecture seule.

**événement distant**

Contrôle les événements distants. Cet attribut est en lecture seule.

**nom de référentiel**

Nom de référentiel. Cet attribut est en lecture seule.

**Liste de noms du référentiel**

Nom de la liste de noms de référentiel. Cet attribut est en lecture seule.

**nom du gestionnaire de files d'attente partagées**

Indique si les MQOPENS d'une file d'attente partagée dont le nom ObjectQMgrest un autre gestionnaire de files d'attente du groupe de partage de files d'attente doivent être résolus en une ouverture de la file d'attente partagée sur le gestionnaire de files d'attente local. Cet attribut est en lecture seule.

**événement ssl**

Indique si des événements SSL sont générés. Cet attribut est en lecture seule.

**FIPS SSL requis**

Indique si seuls les algorithmes certifiés FIPS doivent être utilisés si la cryptographie est exécutée dans le logiciel IBM MQ . Cet attribut est en lecture seule.

**Nbre avant réinit clé SSL**

Nombre d'octets non chiffrés envoyés et reçus dans une conversation SSL avant la renégociation de la clé secrète. Cet attribut est en lecture seule.

**événement de démarrage / arrêt**

Contrôle les événements de démarrage et d'arrêt. Cet attribut est en lecture seule.

**intervalle de collecte de statistiques**

Fréquence à laquelle les données de surveillance des statistiques sont écrites dans la file d'attente de surveillance. Cet attribut est en lecture seule.

**Prise en charge points sync.**

Disponibilité de la participation au point de synchronisation. Cet attribut est en lecture seule.

**Remarque :** Les unités d'oeuvre globales coordonnées par le gestionnaire de files d'attente ne sont pas prises en charge sur la plateforme IBM i . **IBM i** Vous pouvez programmer une unité de travail, coordonnée en externe par IBM i, à l'aide des appels système natifs \_Rcommit et \_Rback. Démarrez ce type d'unité d'oeuvre en démarrant l'application IBM MQ sous contrôle de validation de niveau travail à l'aide de la commande STRCMTCTL. Pour plus d'informations, voir [Interfaces du IBM i gestionnaire de points de synchronisation externe](#) . L'annulation et la validation sont prises en charge sur la plateforme IBM i pour les unités d'oeuvre locales coordonnées par un gestionnaire de files d'attente.

**canaux tcp**

Nombre maximal de canaux pouvant être en cours ou de clients pouvant être connectés, qui utilisent le protocole de transmission TCP/IP. Cet attribut est en lecture seule.

**Conservation de TC**

Indique si la fonction TCP KEEPALIVE doit être utilisée pour vérifier que l'autre extrémité de la connexion est toujours disponible. Cet attribut est en lecture seule.

**Nom TCP**

Nom du système TCP/IP unique ou par défaut à utiliser, en fonction de la valeur du type de pile tcp. Cet attribut est en lecture seule.

**Type de pile TCP**

Indique si l'initiateur de canal est autorisé à utiliser uniquement l'espace adresse TCP/IP indiqué dans le nom tcp ou s'il peut être lié à une adresse TCP/IP sélectionnée. Cet attribut est en lecture seule.

**Enregistrement trace route**

Contrôle l'enregistrement des informations de trace de route. Cet attribut est en lecture seule.

**Intervalle de déclenchement**

Intervalle de déclenchement. Cet attribut est en lecture seule.

**ID utilisateur**

Sur les plateformes AIX and Linux , ID utilisateur réel de l'application. Sur les plateformes Windows , ID utilisateur de l'application.



## Constructeurs

**ImqQueueManager () ;**

Constructeur par défaut.

**ImqQueueManager (const ImqQueueManager & manager) ;**

Constructeur de copie. Le statut de la connexion sera FALSE.

**ImqQueueManager (const char \* name) ;**

Définit le nom ImqObject sur *name*.

## Destructeurs

Lorsqu'un objet gestionnaire ImqQueue est détruit, il est automatiquement déconnecté.

## Méthodes de classe (public)

**comportement MQLONG statique () ;**

Renvoie le comportement.

**void setBehavior(const MQLONG behavior = 0) ;**

Définit le comportement.

## Méthodes d'objet (public)

**void operator = (const ImqQueueManager & mgr) ;**

Se déconnecte si nécessaire et copie les données d'instance de *mgr*. Le statut de la connexion est FALSE.

**ImqBoolean accountingConnRemplacer (MQLONG & statint) ;**

Fournit une copie de la valeur de substitution des connexions de comptabilité. Elle renvoie TRUE en cas de succès.

**MQLONG accountingConnRemplacer () ;**

Renvoie la valeur de substitution des connexions de comptabilité sans indication d'erreurs possibles.

**ImqBoolean accountingInterval (MQLONG & statint) ;**

Fournit une copie de la valeur d'intervalle de comptabilité. Elle renvoie TRUE en cas de succès.

**MQLONG accountingInterval () ;**

Renvoie la valeur de l'intervalle de comptabilité sans aucune indication d'erreurs possibles.

**ImqBoolean activityRecording (MQLONG & rec) ;**

Fournit une copie de la valeur d'enregistrement de l'activité. Elle renvoie TRUE en cas de succès.

**MQLONG activityRecording () ;**

Renvoie la valeur d'enregistrement d'activité sans aucune indication d'erreurs possibles.

**ImqBoolean adoptNewMCACheck (MQLONG & check) ;**

Fournit une copie de la valeur de vérification de l'adoption d'un nouvel agent MCA. Elle renvoie TRUE en cas de succès.

**MQLONG adoptNewMCACheck () ;**

Renvoie la valeur de vérification de l'adoption d'un nouvel agent MCA sans indication d'erreurs éventuelles.

**ImqBoolean adoptNewMCAType (MQLONG & type) ;**

Fournit une copie du nouveau type d'adoption de MCA. Elle renvoie TRUE en cas de succès.

**MQLONG adoptNewMCAType () ;**

Renvoie le nouveau type d'adoption d'agent MCA sans indication d'erreurs éventuelles.

**QLONG authenticationType () const ;**

Renvoie le type d'authentification.

**void setAuthenticationType (const MQLONG type = MQCSP\_AUTH\_NONE) ;**

Définit le type d'authentification.

**ImqBoolean authorityEvent(événement MQLONG & ) ;**

Fournit une copie de l'état d'activation de l'événement d'autorité. Elle renvoie TRUE en cas de succès.

**MQLONG authorityEvent() ;**

Renvoie l'état d'activation de l'événement de droits d'accès sans aucune indication d'erreurs possibles.

**ImqBoolean backout () ;**

Annule les modifications non validées. Elle renvoie TRUE en cas de succès.

**ImqBoolean begin () ;**

Commence une unité de travail. Les options de début affectent le comportement de cette méthode. Elle renvoie TRUE en cas de réussite, mais elle renvoie également TRUE même si l'appel MQBEGIN sous-jacent renvoie MQRC\_NO\_EXTERNAL\_PARTICIPANTS ou MQRC\_PARTICIPANT\_NOT\_AVAILABLE (qui sont tous deux associés à MQCC\_WARNING).

**MQLONG beginOptions() const ;**

Renvoie les options de début.

**void setBeginOptions (const MQLONG options = MQBO\_NONE) ;**

Définit les options de début.

**ImqBoolean bridgeEvent (MQLONG & événement) ;**

Fournit une copie de la valeur d'événement de pont. Elle renvoie TRUE en cas de succès.

**MQLONG bridgeEvent () ;**

Renvoie la valeur d'événement de pont sans aucune indication d'erreurs possibles.

**ImqBoolean channelAutoDéfinition (MQLONG & valeur) ;**

Fournit une copie de la valeur de définition automatique de canal. Elle renvoie TRUE en cas de succès.

**MQLONG channelAutoDéfinition () ;**

Renvoie la valeur de définition automatique de canal sans aucune indication d'erreurs possibles.

**ImqBoolean channelAutoDefinitionEvent(valeur MQLONG & ) ;**

Fournit une copie de la valeur d'événement de définition automatique de canal. Elle renvoie TRUE en cas de succès.

**MQLONG channelAutoDefinitionEvent() ;**

Renvoie la valeur d'événement de définition automatique de canal sans aucune indication d'erreurs possibles.

**ImqBoolean channelAutoDefinitionExit( ImqString & nom ) ;**

Fournit une copie du nom d'exit de définition automatique de canal. Elle renvoie TRUE en cas de succès.

**ImqString channelAutoDefinitionExit() ;**

Renvoie le nom de l'exit de définition automatique de canal sans indication d'erreurs possibles.

**ImqBoolean channelEvent (MQLONG & événement) ;**

Fournit une copie de la valeur d'événement de canal. Elle renvoie TRUE en cas de succès.

**MQLONG channelEvent() ;**

Renvoie la valeur d'événement de canal sans indication d'erreurs possibles.

**MQLONG channelInitiatorAdapters () ;**

Renvoie la valeur des adaptateurs d'initialisateur de canal sans aucune indication d'erreurs possibles.

**ImqBoolean channelInitiatorAdaptateurs (MQLONG & adaptateurs) ;**

Fournit une copie de la valeur des adaptateurs d'initiateur de canal. Elle renvoie TRUE en cas de succès.

**MQLONG channelInitiatorControl () ;**

Renvoie la valeur de démarrage de l'initiateur de canal sans aucune indication d'erreurs possibles.

**Contrôle ImqBoolean channelInitiator(MQLONG & init) ;**

Fournit une copie de la valeur de démarrage du contrôle d'initiateur de canal. Elle renvoie TRUE en cas de succès.

**MQLONG channelInitiatorDispatchers () ;**

Renvoie la valeur des répartiteurs d'initiateur de canal sans aucune indication d'erreurs possibles.

**ImqBoolean channelInitiatorDispatchers (MQLONG & répartiteurs) ;**

Fournit une copie de la valeur des répartiteurs de l'initiateur de canal. Elle renvoie TRUE en cas de succès.

**MQLONG channelInitiatorTraceAutoStart () ;**

Renvoie la valeur de démarrage automatique de la trace de l'initiateur de canal sans aucune indication d'erreurs possibles.

**ImqBoolean channelInitiatorTraceAutoStart (MQLONG & auto) ;**

Fournit une copie de la valeur de démarrage automatique de la trace de l'initiateur de canal. Elle renvoie TRUE en cas de succès.

**MQLONG channelInitiatorTraceTableTaille () ;**

Renvoie la valeur de taille de la table de trace de l'initiateur de canal sans aucune indication d'erreurs possibles.

**ImqBoolean channelInitiatorTraceTableTaille (MQLONG & size) ;**

Fournit une copie de la valeur de taille de la table de trace de l'initiateur de canal. Elle renvoie TRUE en cas de succès.

**ImqBoolean channelMonitoring (MQLONG & monchl) ;**

Fournit une copie de la valeur de surveillance de canal. Elle renvoie TRUE en cas de succès.

**MQLONG channelMonitoring () ;**

Renvoie la valeur de surveillance de canal sans aucune indication d'erreurs possibles.

**ImqBoolean channelReference( ImqChannel \* & pchannel ) ;**

Fournit une copie de la référence de canal. Si la référence de canal n'est pas valide, définissez *pchannel* sur null. Cette méthode renvoie TRUE en cas de succès.

**ImqChannel \* channelReference() ;**

Renvoie la référence de canal sans aucune indication d'erreurs possibles.

**ImqBoolean setChannelRéférence ( ImqChannel & channel ) ;**

Définit la référence de canal. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean setChannelRéférence ( ImqChannel \* channel = 0 ) ;**

Définit ou réinitialise la référence de canal. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean channelStatistics (MQLONG & statchl) ;**

Fournit une copie de la valeur des statistiques de canal. Elle renvoie TRUE en cas de succès.

**MQLONG channelStatistics () ;**

Renvoie la valeur des statistiques de canal sans aucune indication d'erreurs possibles.

**ImqBoolean characterSet(MQLONG & ccsid) ;**

Fournit une copie du jeu de caractères. Elle renvoie TRUE en cas de succès.

**MQLONG characterSet() ;**

Renvoie une copie du jeu de caractères, sans indication des erreurs possibles.

**MQLONG clientSslKeyResetNombre () const ;**

Renvoie la valeur du nombre de réinitialisations de clé SSL utilisée sur les connexions client.

**void setClientSslKeyResetCount(const MQLONG count) ;**

Définit le nombre de réinitialisations de clé SSL utilisées sur les connexions client.

**ImqBoolean clusterSenderMonitoring (MQLONG & monacsl) ;**

Fournit une copie de la valeur par défaut de surveillance de l'émetteur de cluster. Elle renvoie TRUE en cas de succès.

**MQLONG clusterSenderMonitoring () ;**

Renvoie la valeur par défaut de la surveillance de l'émetteur de cluster sans aucune indication d'erreurs possibles.

**ImqBoolean clusterSenderStatistiques (MQLONG & statacls) ;**

Fournit une copie de la valeur des statistiques de l'émetteur de cluster. Elle renvoie TRUE en cas de succès.

**MQLONG clusterSenderStatistics () ;**

Renvoie la valeur des statistiques de l'émetteur de cluster sans aucune indication d'erreurs possibles.

**ImqBoolean clusterWorkloadDonnées ( ImqString & données ) ;**

Fournit une copie des données d'exit de charge de travail du cluster. Elle renvoie TRUE en cas de succès.

**ImqString clusterWorkloadData () ;**

Renvoie les données d'exit de la charge de travail du cluster sans aucune indication d'erreurs possibles.

**ImqBoolean clusterWorkloadExit ( ImqString & name ) ;**

Fournit une copie du nom de l'exit de charge de travail du cluster. Elle renvoie TRUE en cas de succès.

**ImqString clusterWorkloadExit () ;**

Renvoie le nom de l'exit de charge de travail du cluster sans aucune indication d'erreurs possibles.

**ImqBoolean clusterWorkloadLongueur (MQLONG & longueur) ;**

Fournit une copie de la longueur de la charge de travail du cluster. Elle renvoie TRUE en cas de succès.

**MQLONG clusterWorkloadLength () ;**

Renvoie la longueur de la charge de travail du cluster sans aucune indication d'erreurs possibles.

**ImqBoolean clusterWorkLoadMRU (MQLONG & mru) ;**

Fournit une copie de la valeur des canaux utilisés le plus récemment pour la charge de travail du cluster. Elle renvoie TRUE en cas de succès.

**MQLONG clusterWorkLoadMRU () ;**

Renvoie la valeur de la charge de travail de cluster la plus récemment utilisée pour les canaux sans aucune indication d'erreurs possibles.

**ImqBoolean clusterWorkLoadUseQ (MQLONG & useq) ;**

Fournit une copie de la valeur de la file d'attente d'utilisation de la charge de travail du cluster. Elle renvoie TRUE en cas de succès.

**MQLONG clusterWorkLoadUseQ () ;**

Renvoie la valeur de la file d'attente d'utilisation de la charge de travail du cluster sans aucune indication d'erreurs possibles.

**ImqBoolean commandEvent (MQLONG & événement) ;**

Fournit une copie de la valeur d'événement de commande. Elle renvoie TRUE en cas de succès.

**MQLONG commandEvent () ;**

Renvoie la valeur d'événement de commande sans aucune indication d'erreurs possibles.

**ImqBoolean commandInputQueueName( ImqString & nom ) ;**

Fournit une copie du nom de la file d'attente d'entrée des commandes. Elle renvoie TRUE en cas de succès.

**ImqString commandInputQueueName();**

Renvoie le nom de la file d'attente d'entrée de commandes sans indiquer d'erreurs possibles.

**ImqBoolean commandLevel(MQLONG & niveau) ;**

Fournit une copie du niveau de commande. Elle renvoie TRUE en cas de succès.

**MQLONG commandLevel();**

Renvoie le niveau de commande sans aucune indication d'erreurs possibles.

**MQLONG commandServerControl () ;**

Renvoie la valeur de démarrage du serveur de commandes sans aucune indication d'erreurs possibles.

**ImqBoolean commandServerControl (MQLONG & serveur) ;**

Fournit une copie de la valeur de démarrage du contrôle du serveur de commandes. Elle renvoie TRUE en cas de succès.

**ImqBoolean commit () ;**

Valide les modifications non validées. Elle renvoie TRUE en cas de succès.

**ImqBoolean connect () ;**

Se connecte au gestionnaire de files d'attente avec le nom ImqObject indiqué, la valeur par défaut étant le gestionnaire de files d'attente local. Si vous souhaitez vous connecter à un gestionnaire de files d'attente spécifique, utilisez la méthode ImqObject setName avant la connexion. S'il existe

une référence de canal, elle est utilisée pour transmettre des informations sur la définition de canal à MQCONN dans un MQCD. ChannelType dans MQCD est défini sur MQCHT\_CLNTCONN. Les informations de référence de canal, qui ne sont significatives que pour les connexions client, sont ignorées pour les connexions serveur. Les options de connexion affectent le comportement de cette méthode. Cette méthode définit le statut de la connexion sur TRUE en cas de réussite. Elle renvoie le nouveau statut de connexion.

S'il existe un premier enregistrement d'authentification, la chaîne d'enregistrements d'authentification est utilisée pour authentifier les certificats numériques pour les canaux client sécurisés.

Vous pouvez connecter plusieurs objets de gestionnaire ImqQueueau même gestionnaire de files d'attente. Tous utilisent le même descripteur de connexion MQHCONN et partagent la fonctionnalité UOW pour la connexion associée à l'unité d'exécution. Le premier gestionnaire ImqQueuea se connecter obtient le descripteur MQHCONN. Le dernier gestionnaire ImqQueuea se déconnecter exécute le MQDISC.

Pour un programme à unités d'exécution multiples, il est recommandé d'utiliser un objet ImqQueueManager distinct pour chaque unité d'exécution.

**ImqBinary connectionId () const ;**

Renvoie l'ID de connexion.

**ImqBinary connectionTag () const ;**

Renvoie la balise de connexion.

**ImqBoolean setConnectionTag (const MQBYTE128 tag = 0) ;**

Définit la balise de connexion. Si *tag* est égal à zéro, efface la balise de connexion. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean setConnectionTag (const ImqBinary & tag) ;**

Définit la balise de connexion. La longueur des données de la *balise* doit être égale à zéro (pour effacer la balise de connexion) ou MQ\_CONN\_TAG\_LENGTH. Cette méthode renvoie TRUE en cas de succès.

**MQLONG connectOptions() const ;**

Renvoie les options de connexion.

**void setConnectOptions (const MQLONG options = MQCNO\_NONE) ;**

Définit les options de connexion.

**ImqBoolean connectionStatus() const ;**

Renvoie le statut de la connexion.

**ImqString cryptographicHardware ();**

Renvoie le matériel de cryptographie.

**ImqBoolean setCryptographicMatériel (const char \* hardware = 0) ;**

Définit le matériel de cryptographie. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean deadLetterQueueName( ImqString & nom ) ;**

Fournit une copie du nom de la file d'attente de rebut. Elle renvoie TRUE en cas de succès.

**ImqString deadLetterQueueName();**

Renvoie une copie du nom de la file d'attente de rebut, sans indication d'erreurs possibles.

**ImqBoolean defaultTransmissionQueueName( ImqString & nom ) ;**

Fournit une copie du nom de la file d'attente de transmission par défaut. Elle renvoie TRUE en cas de succès.

**ImqString defaultTransmissionQueueName();**

Renvoie le nom de la file d'attente de transmission par défaut sans indication d'erreurs possibles.

**ImqBoolean disconnect () ;**

Se déconnecte du gestionnaire de files d'attente et définit le statut de la connexion sur FALSE. Ferme tous les objets ImqProcess et ImqQueue associés à cet objet et arrête leur référence de connexion avant de se déconnecter. Si plusieurs objets de gestionnaire ImqQueuesont connectés au même gestionnaire de files d'attente, seule la dernière déconnexion effectue une déconnexion physique ; les

autres effectuent une déconnexion logique. Les modifications non validées sont uniquement validées lors de la déconnexion physique.

Cette méthode renvoie TRUE en cas de succès. S'il est appelé alors qu'il n'y a pas de connexion existante, le code retour est également vrai.

**ImqBoolean distributionLists(MQLONG & prise en charge) ;**

Fournit une copie de la valeur des listes de distribution. Elle renvoie TRUE en cas de succès.

**MQLONG distributionLists() ;**

Renvoie la valeur des listes de distribution sans aucune indication d'erreurs possibles.

**ImqBoolean dnsGroup ( ImqString & group) ;**

Fournit une copie du nom de groupe DNS. Elle renvoie TRUE en cas de succès.

**ImqString dnsGroup () ;**

Renvoie le nom de groupe DNS sans aucune indication d'erreurs possibles.

**ImqBoolean dnsWlm (MQLONG & wlm) ;**

Fournit une copie de la valeur WLM DNS. Elle renvoie TRUE en cas de succès.

**MQLONG dnsWlm () ;**

Renvoie la valeur WLM DNS sans indication des erreurs possibles.

**ImqAuthenticationRecord \* firstAuthenticationRecord () const ;**

Renvoie le premier enregistrement d'authentification.

**void setFirstAuthenticationRecord (const ImqAuthenticationRecord \* air = 0) ;**

Définit le premier enregistrement d'authentification.

**ImqObject \* firstManagedObject () const ;**

Renvoie le premier objet géré.

**ImqBoolean inhibitEvent(événement MQLONG &) ;**

Fournit une copie de l'état d'activation de l'événement d'interdiction. Elle renvoie TRUE en cas de succès.

**MQLONG inhibitEvent() ;**

Renvoie l'état d'activation de l'événement d'interdiction sans aucune indication d'erreurs possibles.

**ImqBoolean ipAddressVersion (MQLONG & version) ;**

Fournit une copie de la valeur de version de l'adresse IP. Elle renvoie TRUE en cas de succès.

**MQLONG ipAddressVersion () ;**

Renvoie la valeur de version de l'adresse IP sans aucune indication d'erreurs possibles.

**ImqBoolean keepAlive (MQLONG & keepalive) ;**

Fournit une copie de la valeur de signal de présence. Elle renvoie TRUE en cas de succès.

**MQLONG keepAlive () ;**

Renvoie la valeur de signal de présence sans aucune indication d'erreurs possibles.

**ImqString keyRepository () ;**

Renvoie le référentiel de clés.

**ImqBoolean setKeyRepository (const char \* repository = 0) ;**

Définit le référentiel de clés. Elle renvoie TRUE en cas de succès.

**ImqBoolean listenerTimer (MQLONG & temporisateur) ;**

Fournit une copie de la valeur du temporisateur du programme d'écoute. Elle renvoie TRUE en cas de succès.

**MQLONG listenerTimer () ;**

Renvoie la valeur du temporisateur du programme d'écoute sans aucune indication d'erreurs possibles.

**ImqBoolean localEvent(événement MQLONG &) ;**

Fournit une copie de l'état d'activation de l'événement local. Elle renvoie TRUE en cas de succès.

**MQLONG localEvent() ;**

Renvoie l'état d'activation de l'événement local sans aucune indication d'erreurs possibles.

**ImqBoolean loggerEvent (MQLONG & nombre) ;**  
Fournit une copie de la valeur d'événement du consignateur. Elle renvoie TRUE en cas de succès.

**MQLONG loggerEvent () ;**  
Renvoie la valeur d'événement du consignateur sans aucune indication d'erreurs possibles.

**ImqBoolean luGroupNom ( ImqString & name) ;**  
Fournit une copie du nom du groupe de LU. Renvoie TRUE si l'opération aboutit

**ImqString luGroupNom () ;**  
Renvoie le nom du groupe de LU sans indiquer d'erreurs possibles.

**ImqBoolean lu62ARMSuffix ( ImqString & suffixe) ;**  
Fournit une copie du suffixe ARM LU62 . Elle renvoie TRUE en cas de succès.

**ImqString lu62ARMSuffix () ;**  
Renvoie le suffixe ARM LU62 sans indication des erreurs possibles

**ImqBoolean luName ( ImqString & name) ;**  
Fournit une copie du nom de LU. Elle renvoie TRUE en cas de succès.

**ImqString luName () ;**  
Renvoie le nom d'unité logique sans indication d'erreurs possibles.

**ImqBoolean maximumActiveCanaux (MQLONG & canaux) ;**  
Fournit une copie de la valeur du nombre maximal de canaux actifs. Elle renvoie TRUE en cas de succès.

**MQLONG maximumActiveChannels () ;**  
Renvoie la valeur du nombre maximal de canaux actifs sans aucune indication d'erreurs possibles.

**ImqBoolean maximumCurrentCanaux (MQLONG & canaux) ;**  
Fournit une copie de la valeur du nombre maximal de canaux en cours. Elle renvoie TRUE en cas de succès.

**MQLONG maximumCurrentChannels () ;**  
Renvoie la valeur du nombre maximal de canaux en cours sans aucune indication d'erreurs possibles.

**ImqBoolean maximumHandles(MQLONG & nombre) ;**  
Fournit une copie du nombre maximal de descripteurs. Elle renvoie TRUE en cas de succès.

**MQLONG maximumHandles() ;**  
Renvoie le nombre maximal de descripteurs sans indication d'erreurs possibles.

**ImqBoolean maximumLu62Channels (MQLONG & canaux) ;**  
Fournit une copie de la valeur maximale des canaux LU62 . Elle renvoie TRUE en cas de succès.

**MQLONG maximumLu62Channels () ;**  
Renvoie la valeur maximale des canaux LU62 sans indication d'erreurs possibles

**ImqBoolean maximumMessageLongueur (MQLONG & longueur) ;**  
Fournit une copie de la longueur maximale des messages. Elle renvoie TRUE en cas de succès.

**MQLONG maximumMessageLongueur () ;**  
Renvoie la longueur maximale des messages sans indication d'erreurs possibles.

**ImqBoolean maximumPriority(MQLONG & priorité) ;**  
Fournit une copie de la priorité maximale. Elle renvoie TRUE en cas de succès.

**MQLONG maximumPriority() ;**  
Renvoie une copie de la priorité maximale, sans indication des erreurs possibles.

**ImqBoolean maximumTcpCanaux (MQLONG & canaux) ;**  
Fournit une copie de la valeur du nombre maximal de canaux TCP. Elle renvoie TRUE en cas de succès.

**MQLONG maximumTcpChannels () ;**  
Renvoie la valeur du nombre maximal de canaux TCP sans indication d'erreurs possibles.

**ImqBoolean maximumUncommittedMessages (MQLONG & nombre) ;**  
Fournit une copie du nombre maximal de messages non validés. Elle renvoie TRUE en cas de succès.

**MQLONG maximumUncommittedMessages () ;**  
Renvoie le nombre maximal de messages non validés sans indication d'erreurs possibles.

**ImqBoolean mqiAccounting (MQLONG & statint) ;**

Fournit une copie de la valeur de comptabilité MQI. Elle renvoie TRUE en cas de succès.

**MQLONG mqiAccounting () ;**

Renvoie la valeur de comptabilité MQI sans aucune indication d'erreurs possibles.

**ImqBoolean mqiStatistics (MQLONG & statmqi) ;**

Fournit une copie de la valeur des statistiques MQI. Elle renvoie TRUE en cas de succès.

**MQLONG mqiStatistics () ;**

Renvoie la valeur des statistiques MQI sans aucune indication d'erreurs possibles.

**ImqBoolean outboundPortMax (MQLONG & max) ;**

Fournit une copie de la valeur maximale du port de communications sortantes. Elle renvoie TRUE en cas de succès.

**MQLONG outboundPortMax () ;**

Renvoie la valeur maximale du port de communications sortantes sans aucune indication d'erreurs possibles.

**ImqBoolean outboundPortMin (MQLONG & min) ;**

Fournit une copie de la valeur minimale du port de communications sortantes. Elle renvoie TRUE en cas de succès.

**MQLONG outboundPortMin () ;**

Renvoie la valeur minimale du port de communications sortantes sans aucune indication d'erreurs possibles.

**Mot de passe ImqBinary () const ;**

Renvoie le mot de passe utilisé sur les connexions client.

**ImqBoolean setPassword (const ImqString & password) ;**

Définit le mot de passe utilisé sur les connexions client.

**ImqBoolean setPassword (const char \* = 0 mot de passe) ;**

Définit le mot de passe utilisé sur les connexions client.

**ImqBoolean setPassword (const ImqBinary & password) ;**

Définit le mot de passe utilisé sur les connexions client.

**ImqBoolean performanceEvent(événement MQLONG & ) ;**

Fournit une copie de l'état d'activation de l'événement de performance. Elle renvoie TRUE en cas de succès.

**MQLONG performanceEvent() ;**

Renvoie l'état d'activation de l'événement de performance sans aucune indication d'erreurs possibles.

**Plateforme ImqBoolean (plateforme MQLONG & ) ;**

Fournit une copie de la plateforme. Elle renvoie TRUE en cas de succès.

**plateforme MQLONG () ;**

Renvoie la plateforme sans aucune indication d'erreurs possibles.

**ImqBoolean queueAccounting (MQLONG & acctq) ;**

Fournit une copie de la valeur de comptabilité de la file d'attente. Elle renvoie TRUE en cas de succès.

**MQLONG queueAccounting () ;**

Renvoie la valeur de comptabilité de la file d'attente sans indication d'erreurs possibles.

**ImqBoolean queueMonitoring (MQLONG & monq) ;**

Fournit une copie de la valeur de surveillance de file d'attente. Elle renvoie TRUE en cas de succès.

**MQLONG queueMonitoring () ;**

Renvoie la valeur de surveillance de file d'attente sans aucune indication d'erreurs possibles.

**ImqBoolean queueStatistics (MQLONG & statq) ;**

Fournit une copie de la valeur des statistiques de file d'attente. Elle renvoie TRUE en cas de succès.

**MQLONG queueStatistics () ;**

Renvoie la valeur des statistiques de file d'attente sans indication d'erreurs possibles.



**ImqBoolean receiveTimeout (MQLONG & délai d'attente) ;**

Fournit une copie de la valeur de délai d'attente de réception. Elle renvoie TRUE en cas de succès.

**MQLONG receiveTimeout () ;**

Renvoie la valeur du délai d'attente de réception sans aucune indication d'erreurs possibles.

**ImqBoolean receiveTimeoutMin (MQLONG & min) ;**

Fournit une copie de la valeur de délai d'attente de réception minimale. Elle renvoie TRUE en cas de succès.

**MQLONG receiveTimeoutMin () ;**

Renvoie la valeur de délai d'attente de réception minimale sans aucune indication d'erreurs possibles.

**ImqBoolean receiveTimeoutType (MQLONG & type) ;**

Fournit une copie du type de délai d'attente de réception. Elle renvoie TRUE en cas de succès.

**MQLONG receiveTimeoutType () ;**

Renvoie le type de délai d'attente de réception sans indication d'erreurs possibles.

**ImqBoolean remoteEvent(événement MQLONG & ) ;**

Fournit une copie de l'état d'activation de l'événement distant. Elle renvoie TRUE en cas de succès.

**MQLONG remoteEvent() ;**

Renvoie l'état d'activation de l'événement distant sans aucune indication d'erreurs possibles.

**ImqBoolean repositoryName( ImqString & nom ) ;**

Fournit une copie du nom du référentiel. Elle renvoie TRUE en cas de succès.

**ImqString repositoryName() ;**

Renvoie le nom du référentiel sans aucune indication d'erreurs possibles.

**ImqBoolean repositoryNameListNom ( ImqString & nom ) ;**

Fournit une copie du nom de la liste de noms de référentiel. Elle renvoie TRUE en cas de succès.

**ImqString repositoryNameListNom () ;**

Renvoie une copie du nom de la liste de noms de référentiel sans indication d'erreurs possibles.

**ImqBoolean sharedQueueQueueManagerNom (MQLONG & nom) ;**

Fournit une copie de la valeur du nom du gestionnaire de files d'attente partagées. Elle renvoie TRUE en cas de succès.

**MQLONG sharedQueueQueueManagerNom () ;**

Renvoie la valeur du nom du gestionnaire de files d'attente partagées sans indiquer les erreurs possibles.

**ImqBoolean sslEvent (MQLONG & événement) ;**

Fournit une copie de la valeur d'événement SSL. Elle renvoie TRUE en cas de succès.

**MQLONG sslEvent () ;**

Renvoie la valeur d'événement SSL sans aucune indication d'erreurs possibles.

**ImqBoolean sslFips (MQLONG & sslfips) ;**

Fournit une copie de la valeur SSL FIPS. Elle renvoie TRUE en cas de succès.

**MQLONG sslFips () ;**

Renvoie la valeur SSL FIPS sans indication d'erreurs possibles.

**ImqBoolean sslKeyResetCount (MQLONG & count) ;**

Fournit une copie de la valeur du nombre de réinitialisations de clé SSL. Elle renvoie TRUE en cas de succès.

**MQLONG sslKeyResetCount () ;**

Renvoie la valeur du nombre de réinitialisations de clé SSL sans indication d'erreurs possibles.

**ImqBoolean startStopÉvénement (MQLONG & événement) ;**

Fournit une copie de l'état d'activation de l'événement de démarrage / arrêt. Elle renvoie TRUE en cas de succès.

**MQLONG startStopÉvénement () ;**

Renvoie l'état d'activation de l'événement de démarrage / arrêt sans aucune indication d'erreurs possibles.

**ImqBoolean statisticsInterval (MQLONG & statint) ;**

Fournit une copie de la valeur d'intervalle de statistiques. Elle renvoie TRUE en cas de succès.

**MQLONG statisticsInterval () ;**

Renvoie la valeur d'intervalle des statistiques sans aucune indication d'erreurs possibles.

**ImqBoolean syncPointDisponibilité (MQLONG & sync) ;**

Fournit une copie de la valeur de disponibilité du point de synchronisation. Elle renvoie TRUE en cas de succès.

**MQLONG syncPointAvailability () ;**

Renvoie une copie de la valeur de disponibilité du point de synchronisation, sans aucune indication d'erreurs possibles.

**ImqBoolean tcpName ( ImqString & nom) ;**

Fournit une copie du nom du système TCP. Elle renvoie TRUE en cas de succès.

**ImqString tcpName () ;**

Renvoie le nom du système TCP sans aucune indication d'erreurs possibles.

**ImqBoolean tcpStackType (MQLONG & type) ;**

Fournit une copie du type de pile TCP. Elle renvoie TRUE en cas de succès.

**MQLONG tcpStackType () ;**

Renvoie le type de pile TCP sans indication d'erreurs possibles.

**ImqBoolean traceRouteRecording (MQLONG & routerec) ;**

Fournit une copie de la valeur d'enregistrement de la route de trace. Elle renvoie TRUE en cas de succès.

**MQLONG traceRouteEnregistrement () ;**

Renvoie la valeur d'enregistrement de la route de trace sans aucune indication d'erreurs possibles.

**ImqBoolean triggerInterval(MQLONG & intervalle) ;**

Fournit une copie de l'intervalle de déclenchement. Elle renvoie TRUE en cas de succès.

**MQLONG triggerInterval() ;**

Renvoie l'intervalle de déclenchement sans aucune indication d'erreurs possibles.

**ImqBinary userId () const ;**

Renvoie l'ID utilisateur utilisé sur les connexions client.

**ImqBoolean setUserID (const ImqString & id) ;**

Définit l'ID utilisateur utilisé sur les connexions client.

**ImqBoolean setUserID (const char \* = 0 id) ;**

Définit l'ID utilisateur utilisé sur les connexions client.

**ImqBoolean setUserId (const ImqBinary & id) ;**

Définit l'ID utilisateur utilisé sur les connexions client.

**Méthodes d'objet (protégées)****void setFirstManagedObject (const ImqObject \* object = 0) ;**

Définit le premier objet géré.

**Données d'objet (protégées)****MQHCONN ohconn**

Descripteur de connexion IBM MQ (significatif uniquement lorsque le statut de la connexion est TRUE).

**Codes raison**

- MQRC\_ATTRIBUT\_LOCKED
- MQRC\_ENVIRONMENT\_ERROR
- MQRC\_FUNCTION\_NOT\_SUPPORTED

- ERREUR MQRC\_REFERENCE\_ERROR
- (codes anomalie pour MQBACK)
- (codes anomalie pour MQBEGIN)
- (codes anomalie pour MQCMIT)
- (codes anomalie pour MQCONNX)
- (codes anomalie pour MQDISC)
- (codes anomalie pour MQCONN)

## Classe C++ d'en-tête ImqReference

Cette classe encapsule les fonctions de la structure de données MQRMH.

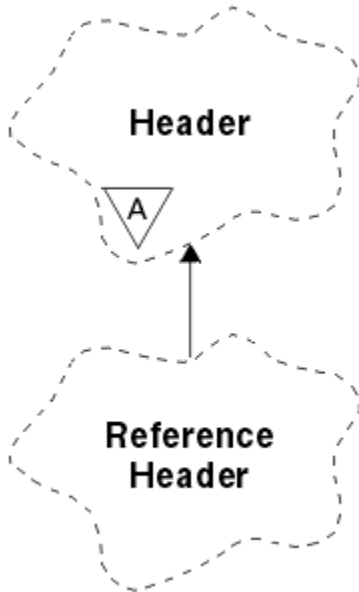


Figure 34. Classe d'en-tête ImqReference

Cette classe est liée aux appels MQI répertoriés dans [«Référence croisée d'en-tête ImqReference»](#), à la page 1887.

- [«Attributs d'objet»](#), à la page 1971
- [«Constructeurs»](#), à la page 1972
- [«Méthodes ImqItem surchargées»](#), à la page 1972
- [«Méthodes d'objet \(public\)»](#), à la page 1972
- [«Données d'objet \(protégées\)»](#), à la page 1973
- [«Codes raison»](#), à la page 1973

### Attributs d'objet

#### environnement de destination

Environnement de la destination. La valeur initiale est une chaîne nulle.

#### nom de destination

Nom de la destination de données. La valeur initiale est une chaîne nulle.

#### id d'instance

Identificateur de l'instance. Une valeur binaire (MQBYTE24) de longueur MQ\_OBJECT\_INSTANCE\_ID\_LENGTH. La valeur initiale est MQOII\_NONE.

**longueur logique**

Longueur logique, ou prévue, des données de message qui suivent cet en-tête. La valeur initiale est zéro.

**décalage logique**

Décalage logique des données de message qui suivent, à interpréter dans le contexte de l'ensemble des données, à la destination finale. La valeur initiale est zéro.

**décalage logique 2**

Extension de haut niveau du décalage logique. La valeur initiale est zéro.

**type de référence**

Type de référence. La valeur initiale est une chaîne nulle.

**environnement source**

Environnement de la source. La valeur initiale est une chaîne nulle.

**nom de source**

Nom de la source de données. La valeur initiale est une chaîne nulle.

**Constructeurs****ImqReferenceEn-tête () ;**

Constructeur par défaut.

**ImqReferenceHeader (const ImqReferenceHeader & header) ;**

Constructeur de copie.

**Méthodes ImqItem surchargées****virtual ImqBoolean copyOut ( ImqMessage & msg ) ;**

Insère une structure de données MQRMH dans la mémoire tampon de message au début, en déplaçant les données de message existantes plus loin, et définit le format *msg* sur MQFMT\_REF\_MSG\_HEADER.

Voir la description de la méthode de classe ImqHeader sur [«Classe C++ ImqHeader»](#), à la page 1916 pour plus de détails.

**virtuel ImqBoolean pasteIn ( ImqMessage & msg ) ;**

Lit une structure de données MQRMH à partir de la mémoire tampon de messages.

Pour que l'opération aboutisse, le format ImqMessage doit être MQFMT\_REF\_MSG\_HEADER.

Voir la description de la méthode de classe ImqHeader sur [«Classe C++ ImqHeader»](#), à la page 1916 pour plus de détails.

**Méthodes d'objet (public)****void operator = (const ImqReferenceHeader & header) ;**

Copie les données d'instance à partir de l'*en-tête*, en remplaçant les données d'instance existantes.

**ImqString destinationEnvironment () const ;**

Renvoie une copie de l'environnement de destination.

**void setDestinationEnvironment (const char \* environment = 0) ;**

Définit l'environnement de destination.

**ImqString destinationName () const ;**

Renvoie une copie du nom de la destination.

**void setDestinationName (const char \* name = 0) ;**

Définit le nom de la destination.

**ImqBinary instanceId () const ;**

Renvoie une copie de l'ID d'instance.

**ImqBoolean setInstanceId (const ImqBinary & id ) ;**

Définit l'ID d'instance. La longueur des données du *jeton* doit être 0 ou MQ\_OBJECT\_INSTANCE\_ID\_LENGTH. Cette méthode renvoie TRUE en cas de succès.

**void setInstanceId (const MQBYTE24 id = 0) ;**

Définit l'ID d'instance. *id* peut être égal à zéro, ce qui revient à spécifier MQOII\_NONE. Si *id* est différent de zéro, il doit traiter les octets MQ\_OBJECT\_INSTANCE\_ID\_LENGTH des données binaires. Lorsque vous utilisez des valeurs prédéfinies telles que MQOII\_NONE, vous devrez peut-être effectuer un transtypage pour garantir une correspondance de signature, par exemple (MQBYTE \*) MQOII\_NONE.

**MQLONG logicalLength () const ;**

Renvoie la longueur logique.

**void setLogicalLength (const MQLONG longueur ) ;**

Définit la longueur logique.

**MQLONG logicalOffset () const ;**

Renvoie le décalage logique.

**void setLogicalOffset (const MQLONG offset ) ;**

Définit le décalage logique.

**MQLONG logicalOffset2 () const ;**

Renvoie le décalage logique 2.

**void setLogicalOffset2 (const MQLONG décalage ) ;**

Définit le décalage logique 2.

**ImqString referenceType () const ;**

Renvoie une copie du type de référence.

**void setReferenceType (const char \* name = 0) ;**

Définit le type de référence.

**ImqString sourceEnvironment () const ;**

Renvoie une copie de l'environnement source.

**void setSourceEnvironment (const char \* environment = 0) ;**

Définit l'environnement source.

**ImqString sourceName () const ;**

Renvoie une copie du nom de la source.

**void setSourceName (const char \* name = 0) ;**

Définit le nom de la source.

**Données d'objet (protégées)****MQRMH omqrmh**

Structure de données MQRMH.

**Codes raison**

- MQRC\_BINARY\_DATA\_LENGTH\_ERROR
- MQRC\_STRUC\_LENGTH\_ERROR
- MQRC\_STRUC\_ID\_ERROR
- MQRC\_DONNÉES\_INSUFFISANTES
- MQRC\_INCONSISTENT\_FORMAT
- ERREUR\_CODAGE\_MQRC\_ERREUR

**Classe C++ ImqString**

Cette classe fournit un stockage et une manipulation de chaînes de caractères pour les chaînes à terminaison nulle.

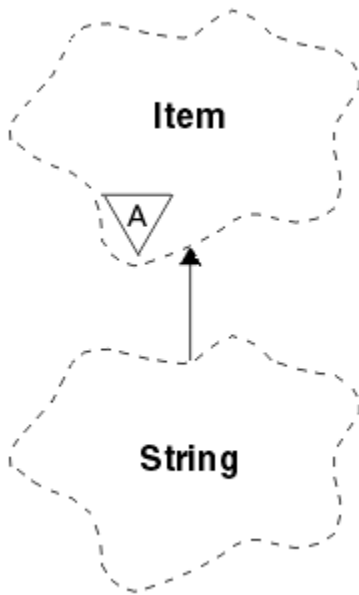


Figure 35. Classe ImqString

Utilisez un ImqString à la place d'un **char \*** dans la plupart des cas où un paramètre appelle un **char \***.

- «Attributs d'objet», à la page 1974
- «Constructeurs», à la page 1974
- «Méthodes de classe (public)», à la page 1975
- «Méthodes ImqItem surchargées», à la page 1975
- «Méthodes d'objet (public)», à la page 1975
- «Méthodes d'objet (protégées)», à la page 1978
- «Codes raison», à la page 1978

## Attributs d'objet

### caractères

Caractères du **stockage** qui précèdent une valeur nulle de fin.

### length

Nombre d'octets dans les **caractères**. S'il n'y a pas de **stockage**, la **longueur** est égale à zéro. La valeur initiale est zéro.

### stockage

Tableau volatil d'octets de taille arbitraire. Une valeur nulle de fin doit toujours être présente dans le **stockage** après les **caractères**, afin que la fin des **caractères** puisse être détectée. Les méthodes garantissent que cette situation est maintenue, mais s'assurent, lors de la définition directe des octets dans le tableau, qu'une valeur nulle de fin existe après la modification. Initialement, il n'existe pas d'attribut **storage**.

## Constructeurs

### ImqString();

Constructeur par défaut.

### ImqString(const ImqString & chaîne);

Constructeur de copie.

### ImqString(const char c);

Les **caractères** comprennent c.

**ImqString(const char \* *texte* ) ;**

Les **caractères** sont copiés à partir de *texte*.

**ImqString(const void \* *buffer*, const size\_t *longueur* ) ;**

Copie *length* octets à partir de *buffer* et les affecte aux **caractères**. La substitution est effectuée pour tous les caractères nuls copiés. Le caractère de substitution est un point (.). Aucune attention particulière n'est accordée aux autres caractères non imprimables ou non affichables copiés.

## Méthodes de classe (public)

**static ImqBoolean copy (char \* *destination-buffer*, const size\_t *longueur*, const char \* *source-buffer*, const char *pad* = 0) ;**

Copie jusqu'à *longueur* octets de *source-buffer* vers *destination-buffer*. Si le nombre de caractères dans *source-buffer* est insuffisant, l'espace restant dans *destination-buffer* est rempli avec les caractères *pad* . La valeur de *source-buffer* peut être zéro. *destination-buffer* peut être égal à zéro si *length* est également égal à zéro. Tous les codes d'erreur sont perdus. Cette méthode renvoie TRUE en cas de succès.

**static ImqBoolean copy (char \* *destination-buffer*, const size\_t *length*, const char \* *source-buffer*, ImqError & *error-object*, const char *pad* = 0) ;**

Copie jusqu'à *longueur* octets de *source-buffer* vers *destination-buffer*. Si le nombre de caractères dans *source-buffer* est insuffisant, l'espace restant dans *destination-buffer* est rempli avec les caractères *pad* . La valeur de *source-buffer* peut être zéro. *destination-buffer* peut être égal à zéro si *length* est également égal à zéro. Tous les codes d'erreur sont définis dans *error-object*. Cette méthode renvoie TRUE en cas de succès.

## Méthodes ImqItem surchargées

**virtual ImqBoolean copyOut ( ImqMessage & *msg* ) ;**

Copie les **caractères** dans la mémoire tampon de messages, en remplaçant tout contenu existant. Définit le **format** *msg* sur MQFMT\_STRING.

Pour plus de détails, voir la description de la méthode de la classe parent.

**virtuel ImqBoolean pasteIn ( ImqMessage & *msg* ) ;**

Définit les **caractères** en transférant les données restantes de la mémoire tampon de messages, en remplaçant les **caractères** existants.

Pour que l'opération aboutisse, le **codage** de l'objet *msg* doit être MQENC\_NATIVE. Extrayez les messages avec MQGMO\_CONVERT en MQENC\_NATIVE.

Pour que l'opération aboutisse, le **format** ImqMessage doit être MQFMT\_STRING.

Pour plus de détails, voir la description de la méthode de la classe parent.

## Méthodes d'objet (public)

**char & operator [ ] (const size\_t *décalage* ) const ;**

Fait référence au caractère au décalage *décalage* dans le **stockage**. Vérifiez que l'octet approprié existe et qu'il est adressable.

**Opérateur ImqString () (const size\_t *offset*, const size\_t *longueur* = 1) const ;**

Renvoie une sous-chaîne en copiant des octets à partir des **caractères** à partir de *offset*. Si *length* est égal à zéro, renvoie le reste des **caractères**. Si la combinaison de *offset* et *length* ne génère pas de référence dans les **caractères**, renvoie une valeur ImqStringvide.

**void operator = (const ImqString & *chaîne* ) ;**

Copie les données d'instance à partir de *string*, en remplaçant les données d'instance existantes.

**Opérateur ImqString + (const char *c* ) const ;**

Renvoie le résultat de l'ajout de *c* aux caractères .

**Opérateur ImqString + (const char \* text) const ;**

Renvoie le résultat de l'ajout de *text* aux **caractères**. Ceci peut également être inversé. Exemple :

```
strOne + "string two" ;  
"string one" + strTwo ;
```

**Remarque :** Bien que la plupart des compilateurs acceptent **strOne + "string two"** ; Microsoft Visual C++ requiert **strOne + (char \*) "string two"** ;

**Opérateur ImqString + (const ImqString & string1) const ;**

Renvoie le résultat de l'ajout de *string1* aux caractères .

**Opérateur ImqString + (const double nombre) const ;**

Renvoie le résultat de l'ajout de *nombre* aux **caractères** après la conversion en texte.

**Opérateur ImqString + (const long number) const ;**

Renvoie le résultat de l'ajout de *nombre* aux **caractères** après la conversion en texte.

**void operator + = (const char c) ;**

Ajoute *c* aux **caractères**.

**void operator + = (const char \* texte) ;**

Ajoute *texte* aux **caractères**.

**void operator + = (const ImqString & chaîne) ;**

Ajoute *string* aux **caractères**.

**void operator + = (const double nombre) ;**

Ajoute un *nombre* aux **caractères** après la conversion en texte.

**void operator + = (const long nombre) ;**

Ajoute un *nombre* aux **caractères** après la conversion en texte.

**caractère opérateur \* () const ;**

Renvoie l'adresse du premier octet du **stockage**. Cette valeur peut être égale à zéro et est volatile. Utilisez cette méthode uniquement à des fins de lecture seule.

**Opérateur ImqBoolean < (const ImqString & string) const ;**

Compare les **caractères** à ceux de *string* à l'aide de la méthode **compare** . Le résultat est TRUE si inférieur à et FALSE si supérieur ou égal à.

**Opérateur ImqBoolean > (const ImqString & string) const ;**

Compare les **caractères** à ceux de *string* à l'aide de la méthode **compare** . Le résultat est TRUE si supérieur à et FALSE si inférieur ou égal à.

**Opérateur ImqBoolean < = (const ImqString & string) const ;**

Compare les **caractères** à ceux de *string* à l'aide de la méthode **compare** . Le résultat est TRUE si inférieur ou égal à et FALSE si supérieur à.

**Opérateur ImqBoolean > = (const ImqString & string) const ;**

Compare les **caractères** à ceux de *string* à l'aide de la méthode **compare** . Le résultat est TRUE si supérieur ou égal à et FALSE si inférieur à.

**Opérateur ImqBoolean == (const ImqString & string) const ;**

Compare les **caractères** à ceux de *string* à l'aide de la méthode **compare** . Elle renvoie TRUE ou FALSE.

**Opérateur ImqBoolean != (const ImqString & string) const ;**

Compare les **caractères** à ceux de *string* à l'aide de la méthode **compare** . Elle renvoie TRUE ou FALSE.

**short compare (const ImqString & chaîne) const ;**

Compare les **caractères** à ceux de *string*. Le résultat est zéro si les **caractères** sont égaux, négatif si inférieur à et positif si supérieur à. La comparaison est sensible à la casse. Une valeur ImqString nulle est considérée comme inférieure à une valeur ImqString non nulle.



**ImqBoolean copyOut(car \* buffer, const size\_t length, const char pad = 0) ;**

Copie jusqu'à *longueur* octets des **caractères** dans la mémoire tampon . Si le nombre de **caractères** est insuffisant, remplit l'espace restant dans la *mémoire tampon* avec des caractères *pad* . *buffer* peut être égal à zéro si *length* est également égal à zéro. Elle renvoie TRUE en cas de succès.

**size\_t copyOut(long & nombre) const ;**

Définit *nombre* à partir des **caractères** après la conversion à partir du texte et renvoie le nombre de caractères impliqués dans la conversion. Si cette valeur est égale à zéro, aucune conversion n'a été effectuée et *nombre* n'est pas défini. Une séquence de caractères convertible doit commencer par les valeurs suivantes:

```
<blank(s)>
<+|->
digit(s)
```

**size\_t copyOut( ImqString & jeton, const char c =") const ;**

Si les **caractères** contiennent un ou plusieurs caractères différents de *c*, identifie un jeton comme la première séquence contiguë de ces caractères. Dans ce cas, *token* est défini sur cette séquence et la valeur renvoyée est la somme du nombre de caractères de début *c* et du nombre d'octets dans la séquence. Sinon, renvoie zéro et ne définit pas *token*.

**size\_t cutOut(long & nombre) ;**

Définit *number* comme pour la méthode **copy** , mais supprime également de **characters** le nombre d'octets indiqué par la valeur de retour. Par exemple, la chaîne illustrée dans l'exemple suivant peut être découpée en trois nombres à l'aide de **cutOut** (*nombre*) trois fois:

```
strNumbers = "-1 0 +55 "
while ( strNumbers.cutOut( number ) );
number becomes -1, then 0, then 55
leaving strNumbers == " "
```

**size\_t cutOut( ImqString & jeton, const char c ="**

Définit *token* comme pour la méthode **copyOut** et supprime de **caractères** les caractères *strToken* ainsi que les caractères *c* qui précèdent les caractères *token* . Si *c* n'est pas un blanc, supprime les caractères *c* qui succèdent directement aux caractères *token* . Renvoie le nombre de caractères supprimés. Par exemple, la chaîne illustrée dans l'exemple suivant peut être découpée en trois jetons à l'aide de **cutOut** (*token*) trois fois:

```
strText = " Program Version 1.1 "
while ( strText.cutOut( token ) );
// token becomes "Program", then "Version",
// then "1.1" leaving strText == " "
```

L'exemple suivant montre comment analyser un nom de chemin DOS:

```
strPath = "C:\OS2\BITMAP\OS2LOGO.BMP"
strPath.cutOut( strDrive, ':' );
strPath.stripLeading( ':' );
while ( strPath.cutOut( strFile, '\' ) );
// strDrive becomes "C".
// strFile becomes "OS2", then "BITMAP",
// then "OS2LOGO.BMP" leaving strPath empty.
```

**ImqBoolean find (const ImqString & chaîne) ;**

Recherche une correspondance exacte pour *chaîne* n'importe où dans les **caractères**. Si aucune correspondance n'est trouvée, elle retourne FALSE. Sinon, elle renvoie TRUE. Si *string* est null, elle renvoie TRUE.

**ImqBoolean find (const ImqString & string, size\_t & décalage) ;**

Recherche une correspondance exacte pour *chaîne* quelque part dans les **caractères** à partir du décalage *décalage* . Si *string* est null, elle renvoie TRUE sans mettre à jour *offset*. Si aucune correspondance n'est trouvée, elle renvoie FALSE (la valeur de *offset* peut avoir été augmentée). Si une correspondance est trouvée, elle renvoie TRUE et met à jour *offset* avec le décalage de *string* dans les caractères .

**size\_t length () const ;**

Renvoie la valeur **length**.

**ImqBoolean pasteIn(const double nombre, const char \* format = "%f") ;**

Ajoute un *nombre* aux **caractères** après la conversion en texte. Elle renvoie TRUE en cas de succès.

La spécification *format* est utilisée pour formater la conversion en virgule flottante. S'il est spécifié, il doit être adapté à une utilisation avec **printf** et des nombres à virgule flottante, par exemple **%3f**.

**ImqBoolean pasteIn(const long nombre) ;**

Ajoute un *nombre* aux **caractères** après la conversion en texte. Elle renvoie TRUE en cas de succès.

**ImqBoolean pasteIn(const void \* buffer, const size\_t length) ;**

Ajoute *length* octets de la *mémoire tampon* aux **caractères** et ajoute une valeur null de fin finale. Remplace tous les caractères nuls copiés. Le caractère de substitution est un point (.). Aucune attention particulière n'est accordée aux autres caractères non imprimables ou non affichables copiés. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean set (const char \* buffer, const size\_t longueur) ;**

Définit les **caractères** à partir d'une zone de caractères de longueur fixe, qui peut contenir une valeur null. Ajoute une valeur null aux caractères de la zone de longueur fixe si nécessaire. Cette méthode renvoie TRUE en cas de succès.

**ImqBoolean setStorage(const size\_t longueur) ;**

Alloue (ou réalloue) le **stockage**. Conserve tous les **caractères** d'origine, y compris les caractères nuls de fin, s'il y a encore de la place pour eux, mais n'initialise pas de stockage supplémentaire.

Cette méthode renvoie TRUE en cas de succès.

**size\_t storage () const ;**

Renvoie le nombre d'octets dans le **stockage**.

**size\_t stripLeading(const char c = " ") ;**

Supprime les caractères de début *c* des **caractères** et renvoie le nombre supprimé.

**size\_t stripTrailing(const char c = " ") ;**

Supprime les caractères de fin *c* des **caractères** et renvoie le nombre supprimé.

**ImqString upperCase() const ;**

Renvoie une copie en majuscules des **caractères**.

**Méthodes d'objet (protégées)****ImqBoolean assign (const ImqString & chaîne) ;**

Équivalent à la méthode **operator =** équivalente, mais non virtuelle. Elle renvoie TRUE en cas de succès.

**Codes raison**

- MQRC\_DATA\_TRONQUÉ
- POINT\_NULL\_MQRC\_NULL
- MQRC\_STORAGE\_NOT\_AVAILABLE
- MQRC\_BUFFER\_ERROR
- MQRC\_INCONSISTENT\_FORMAT

## Classe C++ ImqTrigger

Cette classe encapsule la structure de données MQTM (message de déclenchement).

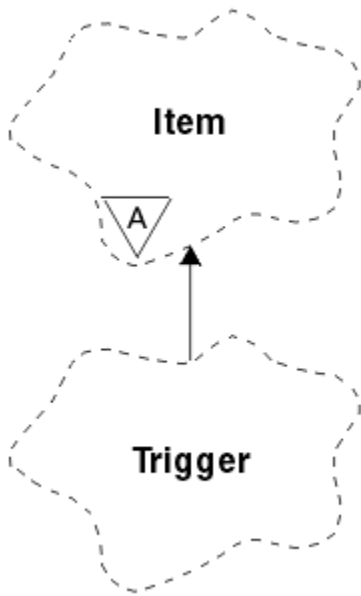


Figure 36. Classe ImqTrigger

Les objets de cette classe sont généralement utilisés par un programme de moniteur de déclenchement. La tâche d'un programme de moniteur de déclenchement consiste à attendre ces messages spécifiques et à les traiter afin de s'assurer que d'autres applications IBM MQ sont démarrées lorsque des messages les attendent.

Pour un exemple d'utilisation, voir l'exemple de programme IMQSTRG.

- [«Attributs d'objet», à la page 1979](#)
- [«Constructeurs», à la page 1980](#)
- [«Méthodes ImqItem surchargées», à la page 1980](#)
- [«Méthodes d'objet \(public\)», à la page 1980](#)
- [«Données d'objet \(protégées\)», à la page 1981](#)
- [«Codes raison», à la page 1981](#)

### Attributs d'objet

#### ID application

Identité de l'application qui a envoyé le message. La valeur initiale est une chaîne nulle.

#### type d'application

Type d'application ayant envoyé le message. La valeur initiale est zéro. Les valeurs supplémentaires suivantes sont possibles:

- MQAT\_AIX
- MQAT\_CICS
- MQAT\_DOS
- MQAT\_IMS
- MQAT\_MVS
- MQAT\_NOTES\_AGENT
- MQAT\_OS2
- MQAT\_OS390

- MQAT\_OS400
- MQAT\_UNIX
- MQAT\_WINDOWS
- MQAT\_WINDOWS\_NT
- MQAT\_USER\_FIRST
- MQAT\_USER\_LAST

#### **données d'environnement**

Données d'environnement du processus. La valeur initiale est une chaîne nulle.

#### **nom de processus**

Nom du processus. La valeur initiale est une chaîne nulle.

#### **nom de la file d'attente**

Nom de la file d'attente à démarrer. La valeur initiale est une chaîne nulle.

#### **données de déclenchement**

Données de déclenchement pour le processus. La valeur initiale est une chaîne nulle.

#### **données utilisateur**

Données utilisateur du processus. La valeur initiale est une chaîne nulle.

### **Constructeurs**

#### **ImqTrigger();**

Constructeur par défaut.

#### **ImqTrigger(const ImqTrigger & *déclencheur* );**

Constructeur de copie.

### **Méthodes ImqItem surchargées**

#### **virtual ImqBoolean copyOut ( ImqMessage & *msg* );**

Écrit une structure de données MQTM dans la mémoire tampon de messages, en remplaçant tout contenu existant. Définit le format *msg* sur MQFMT\_TRIGGER.

Pour plus de détails, voir la description de la méthode de classe ImqItem à l'adresse [«Classe C++ ImqItem»](#), à la page 1920 .

#### **virtuel ImqBoolean pasteIn ( ImqMessage & *msg* );**

Lit une structure de données MQTM à partir de la mémoire tampon de messages.

Pour réussir, le format ImqMessage doit être MQFMT\_TRIGGER.

Pour plus de détails, voir la description de la méthode de classe ImqItem à l'adresse [«Classe C++ ImqItem»](#), à la page 1920 .

### **Méthodes d'objet (public)**

#### **void operator = (const ImqTrigger & *déclencheur* );**

Copie les données d'instance à partir du *déclencheur*, en remplaçant les données d'instance existantes.

#### **ImqString applicationId () const ;**

Renvoie une copie de l'ID d'application.

#### **void setApplicationId (const char \* *id* );**

Définit l'ID de l'application.

#### **MQLONG applicationType () const ;**

Renvoie le type d'application.

#### **void setApplicationType (const MQLONG *type* );**

Définit le type d'application.

**ImqBoolean copyOut ( MQTMC2 \* ptmc2 );**

Encapsule la structure de données MQTM, qui est celle reçue dans les files d'attente d'initialisation. Remplit une structure de données MQTMC2 équivalente fournie par l'appelant et définit la zone QMgrName (qui n'est pas présente dans la structure de données MQTM) sur tous les blancs. La structure de données MQTMC2 est traditionnellement utilisée comme paramètre pour les applications démarrées par un moniteur de déclenchement. Cette méthode renvoie TRUE en cas de succès.

**ImqString environmentData () const ;**

Renvoie une copie des données d'environnement.

**void setEnvironmentData (const char \* data ) ;**

Définit les données d'environnement.

**ImqString processName () const ;**

Renvoie une copie du nom du processus.

**void setProcessName (const char \* name ) ;**

Définit le nom du processus, complété par des blancs, à 48 caractères.

**ImqString queueName () const ;**

Renvoie une copie du nom de la file d'attente.

**void setQueueName (const char \* name ) ;**

Définit le nom de la file d'attente avec des blancs à 48 caractères.

**ImqString triggerData () const ;**

Renvoie une copie des données du déclencheur.

**void setTriggerData (const char \* data ) ;**

Définit les données du déclencheur.

**ImqString userData () const ;**

Renvoie une copie des données utilisateur.

**void setUserData (const char \* data ) ;**

Définit les données utilisateur.

**Données d'objet (protégées)****MQTM omqtm**

Structure de données MQTM.

**Codes raison**

- POINT\_NULL\_MQRC\_NULL
- MQRC\_INCONSISTENT\_FORMAT
- ERREUR\_CODAGE\_MQRC\_ERREUR
- MQRC\_STRUC\_ID\_ERROR

**Classe C++ d'en-tête ImqWork**

Cette classe encapsule des fonctions spécifiques de la structure de données MQWIH.

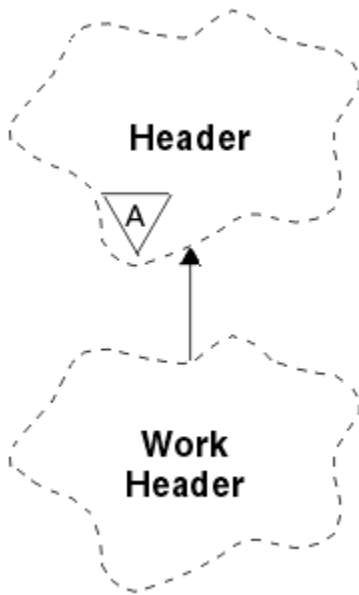


Figure 37. Classe d'en-tête *ImqWork*

Les objets de cette classe sont utilisés par les applications qui placent des messages dans la file d'attente gérée par z/OS Workload Manager.

- «Attributs d'objet», à la page 1982
- «Constructeurs», à la page 1982
- «Méthodes *ImqItem* surchargées», à la page 1982
- «Méthodes d'objet (public)», à la page 1983
- «Données d'objet (protégées)», à la page 1983
- «Codes raison», à la page 1983

## Attributs d'objet

### jeton de message

Jeton de message pour z/OS Workload Manager, de longueur `MQ_MSG_TOKEN_LENGTH`. La valeur initiale est `MQMTOK_NONE`.

### nom de service

Nom de 32 caractères d'un processus. Le nom est initialement à blanc.

### étape de service

Nom de 8 caractères d'une étape dans le processus. Le nom est initialement à blanc.

## Constructeurs

### `ImqWorkHeader ()` ;

Constructeur par défaut.

### `ImqWorkHeader (const ImqWorkHeader & header )` ;

Constructeur de copie.

## Méthodes *ImqItem* surchargées

### `virtual ImqBoolean copyOut( ImqMessage & msg )` ;

Insère une structure de données `MQWIH` au début de la mémoire tampon du message, en déplaçant davantage les données de message existantes et en définissant le format `msg` sur `MQFMT_WORK_INFO_HEADER`.

Pour plus de détails, voir la description de la méthode de classe parent.

### **virtuel ImqBoolean pasteIn( ImqMessage & msg ) ;**

Lit une structure de données MQWIH à partir de la mémoire tampon de messages.

Pour que l'opération aboutisse, le codage de l'objet *msg* doit être MQENC\_NATIVE. Extrayez les messages avec MQGMO\_CONVERT en MQENC\_NATIVE.

Le format ImqMessage doit être MQFMT\_WORK\_INFO\_HEADER.

Pour plus de détails, voir la description de la méthode de classe parent.

## **Méthodes d'objet (public)**

### **void operator = (const ImqWorkHeader & header ) ;**

Copie les données d'instance à partir de l' *en-tête*, en remplaçant les données d'instance existantes.

### **ImqBinary messageToken () const ;**

Renvoie le **jeton de message**.

### **ImqBoolean setMessageToken (const ImqBinary & token ) ;**

Définit le **jeton de message**. La longueur des données du *jeton* doit être égale à zéro ou MQ\_MSG\_TOKEN\_LENGTH. Elle renvoie TRUE en cas de succès.

### **void setMessageToken (const MQBYTE16 token = 0) ;**

Définit le **jeton de message**. La valeur de *token* peut être égale à zéro, ce qui revient à spécifier MQMTOK\_NONE. Si *token* est différent de zéro, il doit traiter les octets MQ\_MSG\_TOKEN\_LENGTH des données binaires.

Lorsque vous utilisez des valeurs prédéfinies telles que MQMTOK\_NONE, vous pouvez être amené à effectuer un transtypage pour garantir une correspondance de signature ; par exemple, (MQBYTE \*) MQMTOK\_NONE.

### **ImqString serviceName () const ;**

Renvoie le **nom de service**, y compris les blancs de fin.

### **void setServiceName (const char \* name ) ;**

Définit le **nom de service**.

### **ImqString serviceStep () const ;**

Renvoie l' **étape de service**, y compris les blancs de fin.

### **void setServiceStep (const char \* étape ) ;**

Définit l' **étape de service**.

## **Données d'objet (protégées)**

### **Omqwih MQWIH**

Structure de données MQWIH.

## **Codes raison**

- MQRC\_BINARY\_DATA\_LENGTH\_ERROR

## **Propriétés des objets IBM MQ classes for JMS**

---

Tous les objets dans IBM MQ classes for JMS ont des propriétés. Différentes propriétés s'appliquent à différents types d'objet. Les différentes propriétés ont des valeurs admises différentes et les valeurs des propriétés symboliques diffèrent entre l'outil d'administration et le code de programme.

IBM MQ classes for JMS fournit des fonctions permettant de définir et d'interroger les propriétés des objets à l'aide de l'outil d'administration de IBM MQ JMS , IBM MQ Explorer, ou dans une application. De nombreuses propriétés ne sont pertinentes que pour un sous-ensemble spécifique des types d'objet.

Pour plus d'informations sur l'utilisation de l'outil d'administration d' IBM MQ JMS , voir [Configuration des objets JMS à l'aide de l'outil d'administration](#).

Tableau 868, à la page 1984 fournit une brève description de chaque propriété et indique, pour chaque propriété, les types d'objet auxquels elle s'applique. Les types d'objet sont identifiés à l'aide de mots clés ; voir Configuration des objets JMS à l'aide de l'outil d'administration pour obtenir une explication de ces objets.

Les numéros se réfèrent aux notes à la fin du tableau. Voir aussi «Dépendances entre les propriétés des objets IBM MQ classes for JMS», à la page 1987.

Une propriété se compose d'une paire nom-valeur au format suivant:

```
PROPERTY_NAME(property_value)
```

Les rubriques de cette section répertorient, pour chaque propriété, le nom de la propriété et une brève description, et affichent les valeurs de propriété valides utilisées dans l'outil d'administration. et la méthode set utilisée pour définir la valeur de la propriété dans une application. Les rubriques affichent également les valeurs de propriété valides pour chaque propriété et le mappage entre les valeurs de propriété symboliques utilisées dans l'outil et leurs équivalents programmables.

Les noms de propriété ne sont pas sensibles à la casse et sont limités à l'ensemble des noms reconnus affichés dans ces rubriques.

Propriété	Forme abrégée	Type d'objet							
		CF	QCF	TCF	Q	T	XACF	XAQCF	Fonction XATCF
«APPLICATIONNAME», à la page 1989	APPNAME	Y	Y	Y			Y	Y	Y
«ASYNCEXCEPTION», à la page 1990	AEX	Y	Y	Y			Y	Y	Y
«OPTIONS balance», à la page 1991	OPTIONS	Y	Y	Y			Y	Y	Y
«TYPE D'ÉQUILIBRAGE», à la page 1991	TYPE	Y	Y	Y			Y	Y	Y
«DÉLAI_ATTENTE», à la page 1992	TIMEOUT	Y	Y	Y			Y	Y	Y
«BROKERCCDURSUBQ», à la page 1992 <sup>1</sup>	CCDSUB					Y			
«BROKERCCSUBQ», à la page 1993 <sup>1</sup>	CCSUB	Y		Y			Y		Y
«BROKERCONQ», à la page 1993 <sup>1</sup>	BCON	Y		Y			Y		Y
«BROKERDURSUBQ», à la page 1994 <sup>1</sup>	BDSUB					Y			
«BROKERPUBQ», à la page 1994 <sup>1</sup>	BPUB	Y		Y		Y	Y		Y
«BROKERPUBQMGR», à la page 1995 <sup>1</sup>	BPQM					Y			
«BROKERQMGR», à la page 1995 <sup>1</sup>	BQM	Y		Y			Y		Y
«BROKERSUBQ», à la page 1995 <sup>1</sup>	BSUB	Y		Y			Y		Y
«BROKERVER», à la page 1996 <sup>1</sup>	BVER	O <sup>2</sup>		O <sup>2</sup>		Y	Y		Y
«CCDTURL», à la page 1997 <sup>3</sup>	Table de définition de canal du client (CCDT)	Y	Y	Y			Y	Y	Y



Tableau 868. Noms de propriété et types d'objet applicables (suite)

Propriété	Forme abrégée	Type d'objet							
		CF	QCF	TCF	Q	T	XACF	XAQCF	Fonction XATCF
«CCSID», à la page 1997	CCS	Y	Y	Y	Y	Y	Y	Y	Y
«Canal», à la page 1998 <sup>3</sup>	CHAN	Y	Y	Y			Y	Y	Y
«CLEANUP», à la page 1998 <sup>1</sup>	CL	Y		Y			Y		Y
«CLEANUPINT», à la page 1999 <sup>1</sup>	CLINT	Y		Y			Y		Y
«ConnectionNameList», à la page 1999	LISTE DES CNTS	Y	Y	Y					
«CLIENTRECONNECTOPTIONS», à la page 1999	CROPT	Y	Y	Y					
«CLIENTRECONNECTTIMEOUT», à la page 2000	CRT	Y	Y	Y					
«IDCLIENT», à la page 2001	CID	O <sup>2</sup>	Y	O <sup>2</sup>			Y	Y	Y
«CLONESUPP», à la page 2001	CLS	Y		Y			Y		Y
«COMPHDR», à la page 2002	pays d'origine	Y		Y			Y		Y
«COMPMSG», à la page 2002	MC	Y	Y	Y			Y	Y	Y
«CONNOPT», à la page 2003	CNOPT	Y	Y	Y			Y	Y	Y
«CONNTAG», à la page 2004	CNTAG	Y	Y	Y			Y	Y	Y
«DESCRIPTION», à la page 2004	DESC	O <sup>2</sup>	Y	O <sup>2</sup>	Y	Y	Y	Y	Y
«DIRECTAUTH», à la page 2005	DAUTH	O <sup>2</sup>		O <sup>2</sup>					
«ENCODING», à la page 2005	ENC				Y	Y			
«EXPIRY», à la page 2006	EXP				Y	Y			
«FAILIFQUIESCE», à la page 2007	FIQ	Y	Y	Y	Y	Y	Y	Y	Y
«HOSTNAME», à la page 2007	HOST	O <sup>2</sup>	Y	O <sup>2</sup>			Y	Y	Y
«LOCALADDRESS», à la page 2008	LA	O <sup>2</sup>	Y	O <sup>2</sup>			Y	Y	Y
«NOM_MAPPE», à la page 2008	MMNST	Y	Y	Y			Y	Y	Y
«MAXBUFFSIZE», à la page 2009	MBSZ	O <sup>2</sup>		O <sup>2</sup>					
«MDREAD», à la page 2009	MDR				Y	Y			
«MDWRITE», à la page 2010	MDW				Y	Y			
«MDMSGCTX», à la page 2011	MDCTX				Y	Y			
«MSGBATCHSZ», à la page 2011 <sup>1</sup>	MBS	Y	Y	Y			Y	Y	Y
«MSGBODY», à la page 2012	MBODY				Y	Y			
«MSGRETENTION», à la page 2012	MRET	Y	Y				Y	Y	
«MSGSELECTION», à la page 2013 <sup>1</sup>	MSEL	Y		Y			Y		Y

Tableau 868. Noms de propriété et types d'objet applicables (suite)

Propriété	Forme abrégée	Type d'objet							
		CF	QCF	TCF	Q	T	XACF	XAQCF	Fonction XATCF
«MULTICAST», à la page 2013	MCAST	O <sup>2</sup>		O <sup>2</sup>		Y			
«OPTIMISTICPUBLICATION», à la page 2014 <sup>1</sup>	OPTPUB	Y		Y					
«OUTCOMENOTIFICATION», à la page 2015 <sup>1</sup>	NOTIFY	Y		Y					
«PERSISTENCE», à la page 2015	PER				Y	Y			
«POLLINGINT», à la page 2016 <sup>1</sup>	PINT	Y	Y	Y			Y	Y	Y
«PORT», à la page 2016	PORT	O <sup>2</sup>	Y	O <sup>2</sup>			Y	Y	Y
«PRIORITY», à la page 2017	PRI				Y	Y			
«PROCESSDURATION», à la page 2017 <sup>1</sup>	PROCDUR	Y		Y					
«PROVIDERVERSION», à la page 2018	PVER	Y	Y	Y			Y	Y	Y
«PROXYHOSTNAME», à la page 2020	PHOST	O <sup>2</sup>		O <sup>2</sup>					
«PROXYPORT», à la page 2021	PPORT	O <sup>2</sup>		O <sup>2</sup>					
«PUBACKINT», à la page 2021 <sup>1</sup>	PAI	Y		Y			Y		Y
«PUTASYNCALLOWED», à la page 2021	PAALD				Y	Y			
«QMANAGER», à la page 2022	QMGR	Y	Y	Y	Y		Y	Y	Y
«QUEUE», à la page 2023	QU				Y				
«READAHEADALLOWED», à la page 2023	RAALD				Y	Y			
«READAHEADCLOSEPOLICY», à la page 2024	RACP				Y	Y			
«RECEIVECCSID», à la page 2024	RCCS				Y	Y			
«RECEIVECONVERSION», à la page 2025	RCNV				Y	Y			
«RECEIVEISOLATION», à la page 2025 <sup>1</sup>	RCVISOL	Y		Y					
«RECEXIT», à la page 2026	RCX	Y	Y	Y			Y	Y	Y
«RECEXITINIT», à la page 2026	RCXI	Y	Y	Y			Y	Y	Y
«REPLYTOSTYLE», à la page 2027	RTOST				Y	Y			
«RESCANINT», à la page 2027 <sup>1</sup>	RINT	Y	Y				Y	Y	
«SECEXIT», à la page 2028	SCX	Y	Y	Y			Y	Y	Y
«SECEXITINIT», à la page 2028	SCXI	Y	Y	Y			Y	Y	Y
«SENDCHECKCOUNT», à la page 2029	SCC	Y	Y	Y			Y	Y	Y

Tableau 868. Noms de propriété et types d'objet applicables (suite)

Propriété	Forme abrégée	Type d'objet							
		CF	QCF	TCF	Q	T	XACF	XAQCF	Fonction XATCF
«SENDEXIT», à la page 2029	SDX	Y	Y	Y			Y	Y	Y
«SENDEXITINIT», à la page 2030	SDXI	Y	Y	Y			Y	Y	Y
«SHARECONVALLOWED», à la page 2030	SCALD	Y	Y	Y			Y	Y	Y
«SPARSESUBS», à la page 2031 <sup>1</sup>	SSUBS	Y		Y					
«SSLCIPHERSUITE», à la page 2032	SCPHS	Y	Y	Y			Y	Y	Y
«SSLCRL», à la page 2032	SCRL	Y	Y	Y			Y	Y	Y
«SSLFIPSREQUIRED», à la page 2032	SFIPS	Y	Y	Y			Y	Y	Y
«SSLPEERNAME», à la page 2033	SPEER	Y	Y	Y			Y	Y	Y
«SSLRESETCOUNT», à la page 2033	SRC	Y	Y	Y			Y	Y	Y
«STATREFRESHINT», à la page 2034 <sup>1</sup>	SRI	Y		Y			Y		Y
«SUBSTORE», à la page 2034 <sup>1</sup>	SS	Y		Y			Y		Y
«SYNCPOINTALLGETS», à la page 2035	SPAG	Y	Y	Y			Y	Y	Y
«TARGCLIENT», à la page 2035	TC				Y	Y			
«TARGCLIENTMATCHING», à la page 2036	TCM	Y	Y				Y	Y	
«TEMPMODEL», à la page 2036	marque	Y	Y				Y	Y	
«TEMPQPREFIX», à la page 2037	TQP	Y	Y				Y	Y	
«TEMPTOPICPREFIX», à la page 2037	TTP	Y		Y			Y		Y
«TOPIC», à la page 2038	TOP					Y			
«TRANSPORT», à la page 2038	TRAN	O <sup>2</sup>	Y	O <sup>2</sup>			Y	Y	Y
«WILDCARDFORMAT», à la page 2039	WCFMT	Y		Y			Y		Y

**Remarque :**

1. Cette propriété peut être utilisée avec la version 70 de IBM MQ classes for JMS, mais elle n'a aucun effet pour une application connectée à un gestionnaire de files d'attente IBM WebSphere MQ 7.0, sauf si la propriété PROVIDERVERSION de la fabrique de connexions est définie sur un numéro de version inférieur à 7.
2. Seules les propriétés BROKERVER, CLIENTID, DESCRIPTION, DIRECTAUTH, HOSTNAME, LOCALADDRESS, MAXBUFFSIZE, MULTICAST, PORT, PROXYHOSTNAME, PROXYPORT et TRANSPORT sont prises en charge pour un objet ConnectionFactory ou TopicConnectionFactory lors de l'utilisation d'une connexion en temps réel à un courtier.
3. Les propriétés CCDURL et CHANNEL d'un objet ne doivent pas être définies simultanément.

## Dépendances entre les propriétés des objets IBM MQ classes for JMS

La validité de certaines propriétés dépend des valeurs particulières d'autres propriétés.

Cette dépendance peut se produire dans les groupes de propriétés suivants:

- Propriétés du client
- Propriétés d'une connexion en temps réel à un courtier
- Chaînes d'initialisation d'exit

#### **Propriétés du client**

Pour une connexion à un gestionnaire de files d'attente, les propriétés suivantes ne sont pertinentes que si TRANSPORT a la valeur CLIENT:

- HOSTNAME
- PORT
- Canal
- LOCALADDRESS
- CCDTURL
- CCSID
- COMPHDR
- COMPMSG
- REEXIT
- REEXITINIT
- SEEXIT
- SEEXITINIT
- SENDEXIT
- SENDEXITINIT
- SHARECONVALLOWED
- SSLCIPHERSUITE
- SSLCRL
- SSLFIPSREQUIRED
- SSLPEERNAME
- SSLRESETCOUNT
- APPLICATIONNAME

Vous ne pouvez pas définir de valeurs pour ces propriétés à l'aide de l'outil d'administration si TRANSPORT a la valeur BIND.

Si TRANSPORT a la valeur CLIENT, la valeur par défaut de la propriété BROKERVER est V1 et la valeur par défaut de la propriété PORT est 1414. Si vous définissez explicitement la valeur de BROKERVER ou PORT, une modification ultérieure de la valeur de TRANSPORT ne remplace pas vos choix.

#### **Propriétés d'une connexion en temps réel à un courtier**

Seules les propriétés suivantes sont pertinentes si TRANSPORT a la valeur DIRECT ou DIRECTION THTTP:

- BROKERVER
- IDCLIENT
- DESCRIPTION
- DIRECTAUTH
- HOSTNAME
- LOCALADDRESS
- MAXBUFFSIZE

- MULTICAST (pris en charge uniquement pour DIRECT)
- PORT
- PROXYHOSTNAME (pris en charge uniquement pour DIRECT)
- PROXYPORT (pris en charge uniquement pour DIRECT)

Si TRANSPORT a la valeur DIRECT ou DIRECTIVE THHTTP, la valeur par défaut de la propriété BROKERVER est V2 et la valeur par défaut de la propriété PORT est 1506. Si vous définissez explicitement la valeur de BROKERVER ou PORT, une modification ultérieure de la valeur de TRANSPORT ne remplace pas vos choix.

### Chaînes d'initialisation d'exit

Ne définissez aucune des chaînes d'initialisation d'exit sans indiquer le nom d'exit correspondant. Les propriétés d'initialisation de l'exit sont les suivantes:

- REEXITINIT
- SEEXITINIT
- SENDEXITINIT

Par exemple, si vous spécifiez REEXITINIT(myString) sans spécifier REEXIT(some.exit.classname), une erreur se produit.

### Référence associée

«TRANSPORT», à la page 2038

Nature d'une connexion à un gestionnaire de files d'attente ou à un courtier.

## APPLICATIONNAME

Une application peut définir un nom identifiant sa connexion au gestionnaire de files d'attente. Ce nom d'application est affiché par la commande **DISPLAY CONN MQSC/PCF** (où la zone est appelée **APPLTAG**) ou dans l'écran **Connexions d'application** de IBM MQ Explorer (où la zone est appelée **App name**).

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : APPLICATIONNAME

Nom abrégé de l'outil d'administration JMS : APPNAME

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setAppNom ()
- MQConnectionFactory.getAppNom ()

### Valeurs

Toute chaîne valide ne dépassant pas 28 caractères. Les noms plus longs sont ajustés pour tenir en supprimant les noms de package de début, si nécessaire. Par exemple, si la classe appelante est com.example.MainApp, le nom complet est utilisé, mais si la classe appelante est com.example.dictionaryAndThesaurus.multilingual.mainApp, le nom multilingual.mainApp est utilisé, car il s'agit de la combinaison la plus longue du nom de classe et du nom de package le plus à droite qui correspond à la longueur disponible.

Si le nom de classe lui-même comporte plus de 28 caractères, il est tronqué pour tenir. Par exemple, com.example.mainApplicationForSecondTestCase devient mainApplicationForSecondTest.

 Sous z/OS, le nom d'API (APPNAME) se trouve dans:

- Le mode de liaison est ignoré s'il est défini et, s'il est défini, il ne peut être défini que sur des blancs.
- Le mode client peut être défini et utilisé.

## ASYNCEXCEPTION

Cette propriété détermine si IBM MQ classes for JMS informe un programme d'écoute des exceptions ( ExceptionListener ) uniquement lorsqu'une connexion est interrompue ou lorsqu'une exception se produit de manière asynchrone sur un appel d'API JMS . Cela s'applique à toutes les connexions créées à partir de cette ConnectionFactory pour lesquelles un ExceptionListener est enregistré.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : ASYNCEXCEPTION

Nom abrégé de l'outil d'administration JMS : AEX

### Accès par programme

Méthodes d'accès set / Getters

- Exceptions MQConnectionFactory.setAsync()
- Exceptions MQConnectionFactory.getAsync()

### Valeurs

#### ASYNCEXCEPTIONS\_ALL

Toutes les exceptions détectées de façon asynchrone, en dehors de la portée d'un appel d'API synchrone, et toutes les exceptions de connexion interrompue sont envoyées au programme d'écoute des exceptions.

*Tableau 869. Toutes les exceptions async: environnements et noms de constante associés*

Environnement	Valeur
JMS Outil d'administration	TOUT
Programmation	WMQCONSTANTS.ASYNC_EXCEPTIONS_ALL = -1
IBM MQ Explorer	Tous

#### ASYNCEXCEPTIONS\_CONNECTIONBROKEN

Seules les exceptions signalant une connexion interrompue sont envoyées au programme d'écoute des exceptions. Toutes les autres exceptions se produisant au cours d'un traitement asynchrone ne sont pas rapportées au programme d'écoute des exceptions ; l'application n'est donc pas informée de ces exceptions. Il s'agit de la valeur par défaut de IBM MQ 8.0.0 Fix Pack 2. Voir [JMS: Modifications du programme d'écoute des exceptions dans IBM MQ 8.0](#).

*Tableau 870. Exceptions indiquant une connexion interrompue: environnements et noms de constante associés*

Environnement	Valeur
JMS Outil d'administration	CONNEXION interrompue
Programmation	WMQCONSTANTS.ASYNC_EXCEPTIONS_CONNECTIONBROKEN = 1
IBM MQ Explorer	Connexion interrompue

La constante supplémentaire suivante est définie:

- Depuis IBM MQ 8.0.0 Fix Pack 2: WMQCONSTANTS.ASYNC\_EXCEPTIONS\_DEFAULT = ASYNC\_EXCEPTIONS\_CONNECTIONBROKEN
- Avant IBM MQ 8.0.0 Fix Pack 2: WMQCONSTANTS.ASYNC\_EXCEPTIONS\_DEFAULT = ASYNC\_EXCEPTIONS\_ALL

### Concepts associés

Exceptions dans IBM MQ classes for JMS

## V 9.3.4 OPTIONS balance

Contrôle la façon dont les applications IBM MQ classes for JMS qui utilisent le transport client sont rééquilibrées dans des clusters uniformes.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory.

Nom long de l'outil d'administration JMS : **BALOPTIONS**

Nom abrégé de l'outil d'administration JMS : **OPTIONS**

### Accès par programme

Méthodes d'accès set / getters

- `MQConnectionFactory.setBalancingOptions()`
- `MQConnectionFactory.getBalancingOptions()`

### Valeurs

#### IGNNONE (Aucun)

Le traitement normal des transactions s'applique et les applications ne doivent pas être déplacées au cours d'une transaction.

Cette valeur est mappée à IBM MQ *BalancingOption* MQBNO\_OPTIONS\_NONE.

#### IGNTRANS

Il peut être demandé aux applications de se déplacer au cours d'une transaction.

Cette valeur est mappée à IBM MQ *BalancingOption* MQBNO\_OPTIONS\_IGNORE\_TRANS.

## V 9.3.4 TYPE D'ÉQUILIBRAGE

Contrôle la façon dont les applications IBM MQ classes for JMS qui utilisent le transport client peuvent être rééquilibrées dans un cluster uniforme.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory.

Nom long de l'outil d'administration JMS : **BALTYPE**

Nom abrégé de l'outil d'administration JMS : **TYPE**

### Accès par programme

Méthodes d'accès set / getters

- `MQConnectionFactory.setBalancingApplicationType()`
- `MQConnectionFactory.getBalancingApplicationType()`

## Valeurs

### SIMPLE

Le traitement par défaut des applications dans un cluster uniforme s'applique.

Cette valeur est mappée à IBM MQ *BalancingOption* MQBNO\_BALTYPE\_SIMPLE.

### REPONSE\_DEMANDE\_DEMANDE

L'application ne sera pas invitée à se reconnecter si un **MQPUT** n'a pas été équilibré par un **MQGET**, sauf si le délai d'attente s'est écoulé.

Cette valeur est mappée à IBM MQ *BalancingOption* MQBNO\_BALTYPE\_REQREP.

## V 9.3.4 DÉLAI\_ATTENTE

Contrôle la façon dont les applications IBM MQ classes for JMS qui utilisent le transport client sont rééquilibrées dans un cluster uniforme.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory.

Nom long de l'outil d'administration JMS : **BALTIMEOUT**

Nom abrégé de l'outil d'administration JMS : **TIMEOUT**

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setBalancingTimeout()
- MQConnectionFactory.getBalancingTimeout()

## Valeurs

### JAMAIS

L'application n'expire jamais à des fins de rééquilibrage dans un cluster uniforme.

Cette valeur est mappée à IBM MQ *BalancingOption* MQBNO\_TIMEOUT\_NEVER.

### IMMEDIATE

L'application expire immédiatement à des fins de rééquilibrage dans un cluster uniforme.

Cette valeur est mappée à IBM MQ *BalancingOption* MQBNO\_TIMEOUT\_IMMEDIATE.

### DEFAULT

L'application dépasse le délai d'attente pour le rééquilibrage dans un cluster uniforme après la période par défaut de 10 secondes.

Cette valeur est mappée à IBM MQ *BalancingOption* MQBNO\_TIMEOUT\_AS\_DEFAULT.

### nn

L'application expire à des fins de rééquilibrage dans un cluster uniforme après une période de *nn* secondes.

*nn* peut être compris entre 1 et 999999999.

## BROKERCCDURSUBQ

Nom de la file d'attente à partir de laquelle les messages d'abonnement durable sont extraits pour un ConnectionConsumer.

### Objets applicables

Topic



Nom long de l'outil d'administration JMS : BROKERCCDURSUBQ

Nom abrégé de l'outil d'administration JMS : CCDSUB

### Accès par programme

Méthodes d'accès set / getters

- MQTopic.setBrokerCCDurSubQueue()
- MQTopic.getBrokerCCDurSubQueue()

### Valeurs

#### **SYSTEM.JMS.D.CC.SUBSCRIBER.QUEUE**

Il s'agit de la valeur par défaut.

Toute chaîne valide

## **BROKERCCSUBQ**

Nom de la file d'attente à partir de laquelle les messages d'abonnement non durable sont extraits pour un ConnectionConsumer.

### Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : BROKERCCSUBQ

Nom abrégé de l'outil d'administration JMS : CCSUB

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setBrokerCCSubQueue()
- MQConnectionFactory.getBrokerCCSubQueue()

### Valeurs

#### **SYSTEM.JMS.ND.CC.SUBSCRIBER.QUEUE**

Il s'agit de la valeur par défaut.

Toute chaîne valide

## **BROKERCONQ**

Nom de la file d'attente de contrôle du courtier.

### Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : BROKERCONQ

Nom abrégé de l'outil d'administration JMS : BCON

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setBrokerControlQueue()
- MQConnectionFactory.getBrokerControlQueue()

## Valeurs

### **SYSTEM.BROKER.CONTROL.QUEUE**

Il s'agit de la valeur par défaut.

**Toute chaîne valide**

## **BROKERDURSUBQ**

Lorsque le IBM MQ classes for JMS est utilisé en mode de migration du fournisseur de messagerie IBM MQ , cette propriété indique le nom de la file d'attente à partir de laquelle les messages d'abonnement durable sont extraits.

### **Objets applicables**

Topic

Nom long de l'outil d'administration JMS : BROKERDURSUBQ

Nom abrégé de l'outil d'administration JMS : BDSUB

### **Accès par programme**

Méthodes d'accès set / getters

- MQTopic.setBrokerDurSubQueue()
- MQTopic.getBrokerDurSubQueue()

## Valeurs

### **SYSTEM.JMS.D.SUBSCRIBER.QUEUE**

Il s'agit de la valeur par défaut.

**Toute chaîne valide**

Démarrage avec SYSTEM.JMS.D

### **Tâches associées**

Configuration de la propriété JMS **PROVIDERVERSION**

## **BROKERPUBQ**

Nom de la file d'attente dans laquelle les messages publiés sont envoyés (file d'attente de flux).

### **Objets applicables**

ConnectionFactory, TopicConnectionFactory, Topic, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : BROKERPUBQ

Nom abrégé de l'outil d'administration JMS : BPUB

### **Accès par programme**

Méthodes d'accès set / getters

- MQConnectionFactory.setBrokerPubQueue
- MQConnectionFactory.getBrokerPubQueue

## Valeurs

### **SYSTEM.BROKER.DEFAULT.STREAM**

Il s'agit de la valeur par défaut.

**Toute chaîne valide**

## **BROKERPUBQMGR**

Nom du gestionnaire de files d'attente propriétaire de la file d'attente dans laquelle les messages publiés sur la rubrique sont envoyés.

### **Objets applicables**

Topic

Nom long de l'outil d'administration JMS : BROKERPUBQMGR

Nom abrégé de l'outil d'administration JMS : BPQM

### **Accès par programme**

Méthodes d'accès set / getters

- MQTopic.setBrokerPubQueueManager()
- MQTopic.getBrokerPubQueueManager()

### **Valeurs**

**null**

Il s'agit de la valeur par défaut.

**Toute chaîne valide**

## **BROKERQMGR**

Nom du gestionnaire de files d'attente sur lequel le courtier s'exécute.

### **Objets applicables**

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : BROKERQMGR

Nom abrégé de l'outil d'administration JMS : BQM

### **Accès par programme**

Méthodes d'accès set / getters

- MQConnectionFactory.setBrokerQueueManager()
- MQConnectionFactory.getBrokerQueueManager()

### **Valeurs**

**null**

Il s'agit de la valeur par défaut.

**Toute chaîne valide**

## **BROKERSUBQ**

Lorsque le IBM MQ classes for JMS est utilisé en mode de migration du fournisseur de messagerie IBM MQ , cette propriété indique le nom de la file d'attente à partir de laquelle les messages d'abonnement non durable sont extraits.

## Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : BROKERSUBQ

Nom abrégé de l'outil d'administration JMS : BSUB

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setBrokerSubQueue()
- MQConnectionFactory.getBrokerSubQueue()

## Valeurs

### SYSTEM.JMS.ND.SUBSCRIBER.QUEUE

Il s'agit de la valeur par défaut.

### Toute chaîne valide

Démarrage avec SYSTEM.JMS.ND

### Tâches associées

Configuration de la propriété JMS **PROVIDERVERSION**

## BROKERVER

Version du courtier utilisé.

## Objets applicables

ConnectionFactory, TopicConnectionFactory, Topic, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : BROKERVER

Nom abrégé de l'outil d'administration JMS : BVER

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setBrokerVersion ()
- MQConnectionFactory.getBrokerVersion ()

## Valeurs

### V1

Pour utiliser un courtier de publication / abonnement IBM MQ ou un courtier IBM MQ Integrator, WebSphere Event Broker, WebSphere Business Integration Event Broker ou WebSphere Business Integration Message Broker en mode compatibilité. Il s'agit de la valeur par défaut si TRANSPORT est défini sur BIND ou CLIENT.

### V2

Pour utiliser un courtier IBM MQ Integrator, WebSphere Event Broker, WebSphere Business Integration Event Broker ou WebSphere Business Integration Message Broker en mode natif. Il s'agit de la valeur par défaut si TRANSPORT est défini sur DIRECT ou DIRECTIVE THHTTP.

### non spécifié

Une fois que le courtier a migré de V6 vers V7, définissez cette propriété de sorte que les en-têtes RFH2 ne soient plus utilisés. Après la migration, cette propriété n'est plus pertinente.

## CCDTURL

Adresse URL (Uniform Resource Locator) identifiant le nom et l'emplacement du fichier contenant la table de définition de canal du client et indiquant comment le fichier est accessible.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : CCDTURL

Nom abrégé de l'outil d'administration JMS : CCDT

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setCCDTURL()
- MQConnectionFactory.getCCDTURL()

### Valeurs

**null**

Il s'agit de la valeur par défaut.

**A Uniform Resource Locator (URL)**

## CCSID

Pour les fabriques de connexions, cette propriété indique l'ID de jeu de caractères codés (CCSID) à utiliser pour les flux de données internes avec le gestionnaire de files d'attente. Pour les destinations, la propriété définit le CCSID à utiliser pour coder les données de chaîne dans MapMessages, StreamMessages et TextMessages placés sur cette destination.

**Remarque :** Il n'est normalement pas nécessaire de modifier cette propriété pour les fabriques de connexions.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, Queue, Topic, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : CCSID

Nom abrégé de l'outil d'administration JMS : CCS

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setCCSID()
- MQConnectionFactory.getCCSID()

### Valeurs

**819**

Valeur par défaut d'une fabrique de connexions.

**1208**

Valeur par défaut d'une destination.

**Tout entier positif**

## Concepts associés

JMS Conversion de message

## Canal

Nom du canal de connexion client utilisé.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : CHANNEL

Nom abrégé de l'outil d'administration JMS : CHAN

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setChannel()
- MQConnectionFactory.getChannel()

### Valeurs

#### **SYSTEM.DEF.SVRCONN**

Il s'agit de la valeur par défaut.

**Toute chaîne valide**

## CLEANUP

Niveau de nettoyage pour les magasins d'abonnement BROKER ou MIGRATE.

### Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : CLEANUP

Nom abrégé de l'outil d'administration JMS : CL

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setCleanupNiveau ()
- MQConnectionFactory.getCleanupNiveau ()

### Valeurs

#### **SECURISEE**

Utilisez un nettoyage sécurisé. Il s'agit de la valeur par défaut.

#### **PROP**

Utilisez un nettoyage sûr, fort ou nul en fonction d'une propriété définie sur la ligne de commande Java .

#### **Aucun**

Ne pas utiliser de nettoyage.

#### **FONDATION**

Utilisez un nettoyage puissant.

## **CLEANUPINT**

Intervalle, en millisecondes, entre les exécutions en arrière-plan de l'utilitaire de nettoyage de publication / abonnement.

### **Objets applicables**

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : CLEANUPINT

Nom abrégé de l'outil d'administration JMS : CLINT

### **Accès par programme**

Méthodes d'accès set / getters

- Intervalle MQConnectionFactory.setCleanup()
- MQConnectionFactory.getCleanupIntervalle ()

### **Valeurs**

**360000**

Il s'agit de la valeur par défaut.

**Tout entier positif**

## **ConnectionNameList**

Liste des noms de connexion TCP/IP. La liste est tentée dans l'ordre, une fois par tentative de reconnexion.

### **Objets applicables**

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS : CONNECTIONNAMELIST

Nom abrégé de l'outil d'administration JMS : CNLIST

### **Accès par programme**

Méthodes d'accès set / getters

- MQConnectionFactory.setconnectionNameListe ()
- MQConnectionFactory.getconnectionNameListe ()

### **Valeurs**

Liste séparée par des virgules de HOSTNAME (PORT). HOSTNAME peut être un nom DNS ou une adresse IP.

La valeur par défaut de PORT est 1414.

## **CLIENTRECONNECTOPTIONS**

Options régissant la reconnexion.

### **Objets applicables**

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS : CLIENTRECONNECTOPTIONS

Nom abrégé de l'outil d'administration JMS : CROPT

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setClientReconnectOptions()
- MQConnectionFactory.getClientReconnectOptions()

## Valeurs

### QMGR

L'application peut se reconnecter au même gestionnaire de files d'attente auquel elle est initialement connectée.

Une erreur avec le code anomalie MQRC\_RECONNECT\_QMID\_MISMATCH est renvoyée si le gestionnaire de files d'attente auquel l'application tente de se connecter, comme indiqué dans la liste des noms de connexion, possède un QMID différent du gestionnaire de files d'attente auquel elle s'est connectée à l'origine.

Utilisez cette valeur si une application peut être reconnectée, mais qu'il existe une affinité entre l'application IBM MQ classes for JMS et le gestionnaire de files d'attente auquel elle a établi une connexion pour la première fois.

Choisissez cette valeur si vous souhaitez qu'une application se reconnecte automatiquement à l'instance de secours d'un gestionnaire de files d'attente à haute disponibilité.

Pour utiliser cette valeur à l'aide d'un programme, utilisez la constante WMQConstants.WMQ\_CLIENT\_RECONNECT\_Q\_MGR.

### ANY

L'application peut se reconnecter à l'un des gestionnaires de files d'attente spécifiés dans la liste de noms de connexion.

Utilisez l'option de reconnexion uniquement s'il n'y a pas d'affinité entre les classes IBM MQ pour l'application JMS et le gestionnaire de files d'attente avec lequel il a initialement établi une connexion.

Pour utiliser cette valeur à partir d'un programme, utilisez la constante WMQConstants.WMQ\_CLIENT\_RECONNECT.

### DESACTIVE

L'application ne sera pas reconnectée.

Pour utiliser cette valeur à l'aide d'un programme, utilisez la constante WMQConstants.WMQ\_CLIENT\_RECONNECT\_DISABLED.

### ADEF

La reconnexion automatique de l'application dépend de la valeur de l'attribut de canal IBM MQ DefReconnect.

Il s'agit de la valeur par défaut.

Pour utiliser cette valeur à partir d'un programme, utilisez la constante WMQConstants.WMQ\_CLIENT\_RECONNECT\_AS\_DEF.

## CLIENTRECONNECTTIMEOUT

Délai avant l'arrêt des tentatives de reconnexion.



## Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS : CLIENTRECONNECTTIMEOUT

Nom abrégé de l'outil d'administration JMS : CRT

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setClientReconnectTimeout()
- MQConnectionFactory.setClientReconnectTimeout()

## Valeurs

Intervalle en secondes. Valeur par défaut: 1800 (30 minutes).

## IDCLIENT

L'identificateur de client permet d'identifier de manière unique la connexion de l'application pour les abonnements durables.

## Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : CLIENTID

Nom abrégé de l'outil d'administration JMS : CID

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.IDsetClient()
- MQConnectionFactory.IDgetClient()

## Valeurs

null

Il s'agit de la valeur par défaut.

**Toute chaîne valide**

## CLONESUPP

Indique si deux instances ou plus du même abonné durable à la rubrique peuvent s'exécuter simultanément.

## Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : CLONESUPP

Nom abrégé de l'outil d'administration JMS : CLS

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory. Prise en charge de setClone()
- MQConnectionFactory. Prise en charge de getClone()

## Valeurs

### DESACTIVE

Une seule instance d'un abonné à une rubrique durable peut être exécutée à la fois. Il s'agit de la valeur par défaut.

### Activée

Deux instances ou plus du même abonné durable à la rubrique peuvent s'exécuter simultanément, mais chaque instance doit s'exécuter dans une machine virtuelle Java (JVM) distincte.

## COMPHDR

Liste des techniques pouvant être utilisées pour compresser les données d'en-tête sur une connexion.

### Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : COMPHDR

Nom abrégé de l'outil d'administration JMS : HC

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setHdrCompList()
- MQConnectionFactory.getHdrCompList()

## Valeurs

### Aucun

Il s'agit de la valeur par défaut.

### SYSTEME

La compression de l'en-tête de message RLE est effectuée.

## COMPMSG

Liste des techniques pouvant être utilisées pour compresser les données de message sur une connexion.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : COMPMSG

Nom abrégé de l'outil d'administration JMS : MC

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setMsgCompList()
- MQConnectionFactory.getMsgCompList()

## Valeurs

### Aucun

Il s'agit de la valeur par défaut.

### Liste d'une ou de plusieurs des valeurs suivantes, séparées par des caractères blancs:

RLE ZLIBFAST ZLIBHIGH

## CONNOPT

Contrôle la façon dont les applications IBM MQ classes for JMS qui utilisent le transport de liaisons se connectent au gestionnaire de files d'attente.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory.

Nom long de l'outil d'administration JMS : CONNOPT

Nom abrégé de l'outil d'administration JMS : CNOPT

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setMQConnectionFactoryOptions ()
- MQConnectionFactory.getMQConnectionFactoryOptions ()

## Valeurs

### STANDARD

La nature de la liaison entre l'application et le gestionnaire de files d'attente dépend de la valeur de l'attribut *DefaultBindType* du gestionnaire de files d'attente. La valeur STANDARD est mappée à IBM MQ *ConnectOption* MQCNO\_STANDARD\_BINDING.

### PARTAGÉ

L'application et l'agent du gestionnaire de files d'attente local s'exécutent dans des unités d'exécution distinctes mais partagent certaines ressources. Cette valeur est mappée à IBM MQ *ConnectOption* MQCNO\_SHARED\_BINDING.

### isolé

L'application et l'agent du gestionnaire de files d'attente local s'exécutent dans des unités d'exécution distinctes et ne partagent aucune ressource. La valeur ISOLÉ est mappée à IBM MQ *ConnectOption* MQCNO\_ISOLATED\_BINDING.

### Fastpath

L'application et l'agent du gestionnaire de files d'attente local s'exécutent dans la même unité d'exécution. Cette valeur est mappée à IBM MQ *ConnectOption* MQCNO\_FASTPATH\_BINDING.

### SERIALQM

L'application demande l'utilisation exclusive de la balise de connexion dans la portée du gestionnaire de files d'attente. Cette valeur est mappée à IBM MQ *ConnectOption* MQCNO\_SERIALIZE\_CONN\_TAG\_Q\_MGR.

### SERIALQSG

L'application demande l'utilisation exclusive de la balise de connexion dans la portée du groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente. La valeur SERIALQSG est mappée à IBM MQ *ConnectOption* MQCNO\_SERIALIZE\_CONN\_TAG\_QSG.

### RESTRICTQM

L'application demande l'utilisation partagée de la balise de connexion, mais il existe des restrictions sur l'utilisation partagée de la balise de connexion dans la portée du gestionnaire de files d'attente. Cette valeur est mappée à IBM MQ *ConnectOption* MQCNO\_RESTRICT\_CONN\_TAG\_Q\_MGR.

## RESTRICTQSG

L'application demande l'utilisation partagée de la balise de connexion, mais il existe des restrictions sur l'utilisation partagée de la balise de connexion dans la portée du groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente. Cette valeur est mappée à IBM MQ *ConnectOption* MQCNO\_RESTRICT\_CONN\_TAG\_QSG.

Pour plus d'informations sur les options de connexion IBM MQ , voir [Connexion à un gestionnaire de files d'attente à l'aide de l'appel MQCONNX](#).

## CONNTAG

Balise que le gestionnaire de files d'attente associe aux ressources mises à jour par l'application dans une unité de travail alors que l'application est connectée au gestionnaire de files d'attente.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : CONNTAG

Nom abrégé de l'outil d'administration JMS : CNTAG

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setConnBalise ()
- MQConnectionFactory.getConnBalise ()

### Valeurs

**Un tableau d'octets de 128 éléments, où chaque élément est 0**

Il s'agit de la valeur par défaut.

**N'importe quelle chaîne**

La valeur est tronquée si elle est supérieure à 128 octets.

## DESCRIPTION

Description de l'objet stocké.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, Queue, Topic, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : DESCRIPTION

Nom abrégé de l'outil d'administration JMS : DESC

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setDescription()
- MQConnectionFactory.getDescription()

### Valeurs

**null**

Il s'agit de la valeur par défaut.

**Toute chaîne valide**

## **DIRECTAUTH**

Indique si l'authentification TLS est utilisée sur une connexion en temps réel à un courtier.

### **Objets applicables**

ConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS : DIRECTAUTH

Nom abrégé de l'outil d'administration JMS : DAUTH

### **Accès par programme**

Méthodes d'accès set / getters

- MQConnectionFactory.setDirectAuth ()
- MQConnectionFactory.getDirectAuth ()

### **Valeurs**

#### **BASIC**

Aucune authentification, authentification par nom d'utilisateur ou authentification par mot de passe. Il s'agit de la valeur par défaut.

#### **CERTIFICATE**

Authentification par certificat de clé publique.

## **ENCODING**

Mode de représentation des données numériques dans le corps d'un message lorsque le message est envoyé à cette destination. La propriété spécifie la représentation d'entiers binaires, d'entiers décimaux condensés et de nombres à virgule flottante.

### **Objets applicables**

File d'attente, Rubrique

Nom long de l'outil d'administration JMS : ENCODING

Nom abrégé de l'outil d'administration JMS : ENC

### **Accès par programme**

Méthodes d'accès set / getters

- MQDestination.setEncoding()
- MQDestination.getEncoding()

### **Valeurs**

#### **propriété ENCODING**

Les valeurs valides que la propriété ENCODING peut prendre sont construites à partir des trois sous-propriétés:

#### **Codage d'entier**

Normale ou inversée

## Codage décimal

Normale ou inversée

## codage en virgule flottante

IEEE normal, IEEE inverse ou z/OS

La propriété ENCODING est exprimée sous la forme d'une chaîne de trois caractères avec la syntaxe suivante:

```
{N|R}{N|R}{N|R|3}
```

Dans cette chaîne:

- N indique normal
- R indique que la valeur est inversée
- 3 indique z/OS
- Le premier caractère représente le *codage d'entier*
- Le second caractère représente le *codage décimal*
- Le troisième caractère représente le *codage en virgule flottante*

Fournit un ensemble de douze valeurs possibles pour la propriété ENCODING .

Il existe une valeur supplémentaire, la chaîne NATIVE, qui définit les valeurs de codage appropriées pour la plateforme Java .

Les exemples suivants montrent des combinaisons valides pour ENCODING:

```
ENCODING (NNR)  
ENCODING (NATIVE)  
ENCODING (RR3)
```

## EXPIRY

Heure à laquelle les messages d'une destination arrivent à expiration.

### Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS : EXPIRATION

Nom abrégé de l'outil d'administration JMS : EXP

### Accès par programme

Méthodes d'accès set / getters

- MQDestination.setExpiry()
- MQDestination.getExpiry()

### Valeurs

#### protocole de publication Atom

L'expiration peut être définie par l'application JMS . Il s'agit de la valeur par défaut.

#### UNLIM

Aucune expiration n'a lieu.

#### 0

Aucune expiration n'a lieu.

**Tout entier positif représentant l'expiration en millisecondes.**

## FAILIFQUIESCE

Cette propriété détermine si les appels à certaines méthodes échouent si le gestionnaire de files d'attente est à l'état de mise au repos ou si une application se connecte à un gestionnaire de files d'attente à l'aide du transport CLIENT et que le canal utilisé par l'application a été mis à l'état de mise au repos, par exemple à l'aide de la commande **STOP CHANNEL** ou **STOP CHANNEL MODE(QUIESCE)** MQSC.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, Queue, Topic, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : FAILIFQUIESCE

Nom abrégé de l'outil d'administration JMS : FIQ

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setFailIfQuiesce()
- MQConnectionFactory.getFailIfQuiesce()

### Valeurs

#### YES

Les appels à certaines méthodes échouent si le gestionnaire de files d'attente est à l'état de mise au repos ou si le canal utilisé pour se connecter à un gestionnaire de files d'attente est en cours de mise au repos. Si une application détecte l'une de ces conditions, elle peut effectuer sa tâche immédiate et fermer la connexion, ce qui permet au gestionnaire de files d'attente ou à l'instance de canal de s'arrêter. Il s'agit de la valeur par défaut.

#### NO

Aucun appel de méthode n'a échoué car le gestionnaire de files d'attente ou le canal utilisé pour se connecter à un gestionnaire de files d'attente est à l'état de mise au repos. Si vous spécifiez cette valeur, une application ne peut pas détecter que le gestionnaire de files d'attente ou le canal est en cours de mise au repos. L'application peut continuer à effectuer des opérations sur le gestionnaire de files d'attente et, par conséquent, empêcher l'arrêt du gestionnaire de files d'attente.

## HOSTNAME

Pour une connexion à un gestionnaire de files d'attente, nom d'hôte ou adresse IP du système sur lequel le gestionnaire de files d'attente s'exécute ou, pour une connexion en temps réel à un courtier, nom d'hôte ou adresse IP du système sur lequel le courtier s'exécute.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : HOSTNAME

Nom abrégé de l'outil d'administration JMS : HOST

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setHostNom ()
- MQConnectionFactory.getHostNom ()

## Valeurs

### ystème hôte local

Il s'agit de la valeur par défaut.

### Toute chaîne valide

## LOCALADDRESS

Pour une connexion à un gestionnaire de files d'attente, cette propriété indique soit l'interface réseau locale à utiliser, soit le port local ou la plage de ports locaux à utiliser.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : LOCALADDRESS

Nom abrégé de l'outil d'administration JMS : LA

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setLocalAdresse ()
- MQConnectionFactory.getLocalAdresse ()

## Valeurs

### "" (chaîne vide)

Il s'agit de la valeur par défaut.

### Chaîne au format [ ip-addr ] [ (low-port [, high port ]) ]

Voici quelques exemples :

192.0.2.0

Le canal se lie à l'adresse 192.0.2.0 en local.

192.0.2.0(1000)

Le canal se lie à l'adresse 192.0.2.0 en local et utilise le port 1000.

192.0.2.0(1000,2000)

Le canal se lie à l'adresse 192.0.2.0 en local et utilise un port compris entre 1000 et 2000.

(1000)

Le canal est lié au port 1000 en local.

(1000,2000)

Le canal se lie à un port compris entre 1000 et 2000 localement.

Vous pouvez spécifier un nom d'hôte à la place d'une adresse IP. Pour une connexion en temps réel à un courtier, cette propriété est pertinente uniquement lorsque la multidiffusion est utilisée et la valeur de la propriété ne doit pas contenir de numéro de port ou de plage de numéros de port. Les seules valeurs valides de la propriété dans ce cas sont null, une adresse IP ou un nom d'hôte.

## NOM\_MAPPE

Permet d'utiliser le style de compatibilité pour les noms d'élément MapMessage .



## Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : MAPNAMESTYLE

Nom abrégé de l'outil d'administration JMS : MNST

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setMapNameStyle()
- MQConnectionFactory.getMapNameStyle()

## Valeurs

### STANDARD

Le format de dénomination d'élément standard `com.ibm.jms.JMSMapMessage` doit être utilisé. Il s'agit de la valeur par défaut qui permet d'utiliser des identificateurs Java non légaux comme nom d'élément.

### Compatible

L'ancien format de dénomination de l'élément `com.ibm.jms.JMSMapMessage` doit être utilisé. Seuls les identificateurs Java légaux peuvent être utilisés comme nom d'élément. Cela n'est nécessaire que si des messages de mappe sont envoyés à une application qui utilise une version de IBM MQ classes for JMS antérieure à 5.3.

## MAXBUFFSIZE

Nombre maximal de messages reçus qui peuvent être stockés dans une mémoire tampon de messages internes en attendant d'être traités par l'application. Cette propriété s'applique uniquement lorsque TRANSPORT a la valeur DIRECT ou DIRECTION THHTTP.

## Objets applicables

ConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS : MAXBUFFSIZE

Nom abrégé de l'outil d'administration JMS : MBSZ

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setMaxBufferSize()
- MQConnectionFactory.getMaxBufferSize()

## Valeurs

### 1000

Il s'agit de la valeur par défaut.

**Tout entier positif**

## MDREAD

Cette propriété détermine si une application JMS peut extraire les valeurs des zones MQMD.

## Objets applicables

Nom long de l'outil d'administration JMS : MDREAD

Nom abrégé de l'outil d'administration JMS : MDR

## Accès par programme

Méthodes d'accès set / getters

- MQDestination.setMQMDReadEnabled()
- MQDestination.getMQMDReadEnabled()

## Valeurs

### NO

Lors de l'envoi de messages, les propriétés JMS\_IBM\_MQMD\* d'un message envoyé ne sont pas mises à jour pour refléter les valeurs de zone mises à jour dans le MQMD. Lors de la réception de messages, aucune des propriétés JMS\_IBM\_MQMD\* n'est disponible sur un message reçu, même si l'émetteur les a définies partiellement ou intégralement. Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez False.

### Oui

Lors de l'envoi de messages, toutes les propriétés JMS\_IBM\_MQMD\* d'un message envoyé sont mises à jour pour refléter les valeurs de zone mises à jour dans le MQMD, y compris les propriétés que l'expéditeur n'a pas définies explicitement. Lors de la réception de messages, toutes les propriétés JMS\_IBM\_MQMD\* sont disponibles sur un message reçu, y compris les propriétés que l'expéditeur n'a pas définies explicitement.

Pour les programmes, utilisez True.

## MDWRITE

Cette propriété détermine si une application JMS peut définir les valeurs des zones MQMD.

## Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS : MDWRITE

Nom abrégé de l'outil d'administration JMS : MDR

## Accès par programme

Méthodes d'accès set / getters

- MQDestination.setMQMDWriteEnabled()
- MQDestination.getMQMDWriteEnabled()

## Valeurs

### NO

Toutes les propriétés JMS\_IBM\_MQMD\* sont ignorées et leurs valeurs ne sont pas copiées dans la structure MQMD sous-jacente. Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez False.

### YES

Les propriétés JMS\_IBM\_MQMD\* sont traitées. Leurs valeurs sont copiées dans la structure sous-jacente.

Pour les programmes, utilisez True.

## **MDMSGCTX**

Niveau de contexte de message à définir par l'application JMS . L'application doit être exécutée avec les droits appropriés pour que cette propriété soit appliquée.

### **Objets applicables**

Nom long de l'outil d'administration JMS : MDMSGCTX

Nom abrégé de l'outil d'administration JMS : MDCTX

### **Accès par programme**

Méthodes d'accès set / getters

- MQDestination.setMQMDMessageContext()
- MQDestination.getMQMDMessageContext()

### **Valeurs**

#### **DEFAULT**

L'appel d'API MQOPEN et la structure MQPMO ne spécifient aucune option de contexte de message explicite. Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez WMQ\_MDCTX\_DEFAULT.

#### **SET\_IDENTITY\_CONTEXT**

L'appel API MQOPEN API spécifie l'option de contexte de message MQOO\_SET\_IDENTITY\_CONTEXT et la structure MQPMO spécifie MQPMO\_SET\_IDENTITY\_CONTEXT.

Pour les programmes, utilisez WMQ\_MDCTX\_SET\_IDENTITY\_CONTEXT.

#### **SET\_ALL\_CONTEXT**

L'appel API MQOPEN API spécifie l'option de contexte de message MQOO\_SET\_ALL\_CONTEXT et la structure MQPMO spécifie MQPMO\_SET\_ALL\_CONTEXT.

Pour les programmes, utilisez WMQ\_MDCTX\_SET\_ALL\_CONTEXT.

## **MSGBATCHSZ**

Nombre maximal de messages à prendre dans une file d'attente dans un paquet lors de l'utilisation de la distribution de messages asynchrones.

### **Objets applicables**

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : MAXBUFFSIZE

Nom abrégé de l'outil d'administration JMS : MBSZ

### **Accès par programme**

Méthodes d'accès set / getters

- MQConnectionFactory.setMsgBatchSize()
- MQConnectionFactory.getMsgBatchSize()

## Valeurs

### 10

Il s'agit de la valeur par défaut.

**Tout entier positif**

## MSGBODY

Détermine si une application JMS accède à l' MQRFH2 d'un message IBM MQ dans le cadre de la charge de message.

### Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS : WMQ\_MESSAGE\_BODY

Nom abrégé de l'outil d'administration JMS : MBODY

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setMessageBodyStyle()
- MQConnectionFactory.getMessageBodyStyle()

## Valeurs

### NON SPECIFIE

Lors de l'envoi, IBM MQ classes for JMS génère ou non et inclut un en-tête MQRFH2 , en fonction de la valeur de WMQ\_TARGET\_CLIENT. Lors de la réception, agit en tant que valeur JMS.

### JMS

Lors de l'envoi, IBM MQ classes for JMS génère automatiquement un en-tête MQRFH2 et l'inclut dans le message IBM MQ .

Lors de la réception, IBM MQ classes for JMS définit les propriétés de message JMS en fonction des valeurs dans MQRFH2 (le cas échéant) ; il ne présente pas MQRFH2 dans le corps du message JMS .

### MQ

Lors de l'envoi, IBM MQ classes for JMS ne génère pas de MQRFH2.

Lors de la réception, IBM MQ classes for JMS présente MQRFH2 dans le corps du message JMS .

## MSGRETENTION

Indique si le consommateur de connexion conserve les messages non distribués dans la file d'attente d'entrée.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory,

Nom long de l'outil d'administration JMS : MSGRETENTION

Nom abrégé de l'outil d'administration JMS : MRET

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setMessageRetention ()
- MQConnectionFactory.getMessageConservation ()

## Valeurs

### Oui

Les messages non distribués restent dans la file d'entrée. Il s'agit de la valeur par défaut.

### Non

Les messages non distribués sont traités en fonction de leurs options de disposition.

## MSGSELECTION

Détermine si la sélection des messages est effectuée par le IBM MQ classes for JMS ou par le courtier. Si TRANSPORT a la valeur DIRECT, la sélection des messages est toujours effectuée par le courtier et la valeur de MSGSELECTION est ignorée. La sélection des messages par le courtier n'est pas prise en charge lorsque BROKERVER a la valeur V1.

### Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : MSGSELECTION

Nom abrégé de l'outil d'administration JMS : MSEL

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setMessageSélection ()
- MQConnectionFactory.getMessageSélection ()

## Valeurs

### CLIENT

La sélection des messages est effectuée par IBM MQ classes for JMS. Il s'agit de la valeur par défaut.

### COURTIER

La sélection de message est faite par le courtier.

## MULTICAST

Permet d'activer la multidiffusion sur une connexion en temps réel à un courtier et, si elle est activée, de spécifier la manière précise dont la multidiffusion est utilisée pour distribuer des messages du courtier à un consommateur de message. La propriété n'a aucun effet sur la façon dont un expéditeur de message envoie des messages à un courtier.

### Objets applicables

ConnectionFactory, TopicConnectionFactory, Rubrique

Nom long de l'outil d'administration JMS : MULTICAST

Nom abrégé de l'outil d'administration JMS : MCAST

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setMulticast()
- MQConnectionFactory.getMulticast()

## Valeurs

### DESACTIVE

Les messages ne sont pas distribués à un consommateur de message via le transport multidiffusion. Il s'agit de la valeur par défaut pour les objets `ConnectionFactory` et `TopicConnectionFactory`.

### ASCF

Les messages sont distribués à un consommateur de message en fonction du paramètre de multidiffusion défini pour la fabrique de connexions associée au consommateur de message. Le paramètre de multidiffusion de la fabrique de connexions est noté lors de la création du consommateur de message. Cette valeur est valide uniquement pour les objets de rubrique. Il s'agit de la valeur par défaut pour les objets de rubrique.

### Activée

Si la rubrique est configurée pour la multidiffusion dans le courtier, les messages sont distribués à un consommateur de message à l'aide du transport de multidiffusion. Une qualité de service fiable est utilisée si la rubrique est configurée pour une multidiffusion fiable.

### FIABLE.

Si la rubrique est configurée pour une multidiffusion fiable dans le courtier, les messages sont distribués au consommateur de messages à l'aide d'un transport multidiffusion avec une qualité de service fiable. Si la rubrique n'est pas configurée pour la multidiffusion fiable, vous ne pouvez pas créer de consommateur de message pour la rubrique.

### NOTR

Si la rubrique est configurée pour la multidiffusion dans le courtier, les messages sont distribués au consommateur de message à l'aide du transport de multidiffusion. Une qualité de service fiable n'est pas utilisée, même si la rubrique est configurée pour une multidiffusion fiable.

## OPTIMISTICPUBLICATION

Cette propriété détermine si IBM MQ classes for JMS renvoie immédiatement le contrôle à un diffuseur de publications qui a publié un message, ou s'il renvoie le contrôle uniquement après avoir terminé tout le traitement associé à l'appel et peut signaler le résultat au diffuseur de publications.

### Objets applicables

`ConnectionFactory`, `TopicConnectionFactory`

Nom long de l'outil d'administration JMS : `OPTIMISTICPUBLICATION`

Nom abrégé de l'outil d'administration JMS : `OPTPUB`

### Accès par programme

Méthodes d'accès set / getters

- `MQConnectionFactory.setOptimisticPublication ()`
- `MQConnectionFactory.getOptimisticPublication ()`

## Valeurs

### NO

Lorsqu'un diffuseur publie un message, IBM MQ classes for JMS ne renvoie pas le contrôle au diffuseur tant qu'il n'a pas terminé tout le traitement associé à l'appel et peut signaler le résultat au diffuseur. Il s'agit de la valeur par défaut.

### YES

Lorsqu'un diffuseur de publications publie un message, IBM MQ classes for JMS renvoie le contrôle au diffuseur de publications immédiatement avant qu'il n'ait terminé tout le traitement associé à l'appel et peut signaler le résultat au diffuseur de publications. IBM MQ classes for JMS signale le résultat uniquement lorsque le diffuseur de publications valide le message.

## OUTCOMENOTIFICATION

Cette propriété détermine si IBM MQ classes for JMS renvoie immédiatement le contrôle à un abonné qui vient d'accuser réception d'un message ou s'il renvoie le contrôle uniquement après avoir terminé tout le traitement associé à l'appel et peut signaler le résultat à l'abonné.

### Objets applicables

ConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS : OUTCOMENOTIFICATION

Nom abrégé de l'outil d'administration JMS : NOTIFY

### Accès par programme

Méthodes d'accès set / getters

- Notification MQConnectionFactory.setOutcome()
- MQConnectionFactory.getOutcomeNotification ()

### Valeurs

#### YES

Lorsqu'un abonné accuse réception ou valide un message, IBM MQ classes for JMS ne renvoie pas le contrôle à l'abonné tant qu'il n'a pas terminé tout le traitement associé à l'appel et peut signaler le résultat à l'abonné. Il s'agit de la valeur par défaut.

#### NO

Lorsqu'un abonné accuse réception ou valide un message, IBM MQ classes for JMS renvoie le contrôle à l'abonné immédiatement, avant qu'il n'ait terminé tout le traitement associé à l'appel et qu'il puisse signaler le résultat à l'abonné.

## PERSISTENCE

Persistence des messages envoyés à une destination.

### Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS : PERSISTENCE

Nom abrégé de l'outil d'administration JMS : PER

### Accès par programme

Méthodes d'accès set / getters

- MQDestination.setPersistence()
- MQDestination.getPersistence()

### Valeurs

#### protocole de publication Atom

La persistance est définie par l'application JMS . Il s'agit de la valeur par défaut.

#### QDEF

La persistance prend la valeur par défaut de la file d'attente.

#### pers

Les messages sont persistants.

## **NON**

Les messages sont non persistants.

## **ÉLEVÉ**

Pour plus d'informations sur l'utilisation de cette valeur, voir [Messages persistantsJMS](#) .

## **POLLINGINT**

Si chaque programme d'écoute de messages d'une session n'a pas de message approprié dans sa file d'attente, il s'agit de l'intervalle maximal, en millisecondes, qui s'écoule avant que chaque programme d'écoute de messages tente à nouveau d'extraire un message de sa file d'attente. Si l'absence de message approprié est fréquemment observée pour l'un quelconque des écouteurs de messages au sein d'une session, envisagez d'augmenter la valeur de cette propriété. Cette propriété est pertinente uniquement si TRANSPORT a la valeur BIND ou CLIENT.

### **Objets applicables**

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : POLLINGINT

Nom abrégé de l'outil d'administration JMS : PINT

### **Accès par programme**

Méthodes d'accès set / getters

- MQConnectionFactory.IntervalsetPolling()
- MQConnectionFactory.getPollingIntervalle ()

### **Valeurs**

#### **5000**

Il s'agit de la valeur par défaut.

#### **Tout entier positif**

## **PORT**

Pour une connexion à un gestionnaire de files d'attente, numéro du port sur lequel le gestionnaire de files d'attente est à l'écoute ou, pour une connexion en temps réel à un courtier, numéro du port sur lequel le courtier est à l'écoute des connexions en temps réel.

### **Objets applicables**

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : PORT

Nom abrégé de l'outil d'administration JMS : PORT

### **Accès par programme**

Méthodes d'accès set / getters

- MQConnectionFactory.setPort()
- MQConnectionFactory.getPort()



## Valeurs

### 1414

Il s'agit de la valeur par défaut si TRANSPORT est défini sur CLIENT.

### 1506

Il s'agit de la valeur par défaut si TRANSPORT est défini sur DIRECT ou DIRECTIVE THHTTP.

**Tout entier positif**

## PRIORITY

Priorité des messages envoyés à une destination.

### Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS : PRIORITY

Nom abrégé de l'outil d'administration JMS : PRI

### Accès par programme

Méthodes d'accès set / getters

- MQDestination.setPriority()
- MQDestination.getPriority()

## Valeurs

### protocole de publication Atom

La priorité est définie par l'application JMS . Il s'agit de la valeur par défaut.

### QDEF

La priorité prend la valeur par défaut de la file d'attente.

### Tout entier compris entre 0 et 9

De la plus faible à la plus élevée.

## PROCESSDURATION

Cette propriété détermine si un abonné garantit le traitement rapide des messages qu'il reçoit avant de renvoyer le contrôle à IBM MQ classes for JMS.

### Objets applicables

ConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS : PROCESSDURATION

Nom abrégé de l'outil d'administration JMS : PROCDUR

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setProcessDurée ()
- MQConnectionFactory.getProcessDurée ()

## Valeurs

### INCONNU

Un abonné ne peut donner aucune garantie quant à la rapidité avec laquelle il peut traiter un message qu'il reçoit. Il s'agit de la valeur par défaut.

### SHORT

Un abonné garantit le traitement rapide des messages qu'il reçoit avant de renvoyer le contrôle à IBM MQ classes for JMS.

## PROVIDERVERSION

Cette propriété différencie les trois modes de fonctionnement de la messagerie IBM MQ : le mode normal du fournisseur de messagerie IBM MQ , le mode normal du fournisseur de messagerie IBM MQ avec restrictions et le mode de migration du fournisseur de messagerie IBM MQ .

Le mode de fournisseur de messagerie IBM MQ normal utilise toutes les fonctions d'un gestionnaire de files d'attente IBM MQ pour implémenter JMS. Ce mode est optimisé pour utiliser l'API et la fonctionnalité de JMS 2.0. Le mode normal du fournisseur de messagerie IBM MQ avec restrictions utilise l'API JMS 2.0 , mais pas les nouvelles fonctions telles que les abonnements partagés, la distribution différée ou l'envoi asynchrone.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : PROVIDERVERSION

Nom abrégé de l'outil d'administration JMS : PVER

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setProviderVersion ()
- MQConnectionFactory.getProviderVersion ()

## Valeurs

Vous pouvez définir la propriété **PROVIDERVERSION** sur l'une des valeurs 8 (mode normal), 7 (mode normal avec restrictions), 6 (mode de migration) ou `unspecified` (valeur par défaut). La valeur que vous spécifiez pour la propriété **PROVIDERVERSION** doit être une chaîne. Vous pouvez spécifier l'option 8, 7 ou 6 dans l'un des formats suivants :

- V.R.M.F
- V.R.M
- V.R
- V

où V, R, M et F sont des entiers strictement positifs. Les valeurs R, M et F sont facultatives ; vous pouvez les utiliser si le contrôle à granularité fine est nécessaire. Par exemple, si vous souhaitez utiliser un niveau **PROVIDERVERSION** de 7, vous pouvez définir **PROVIDERVERSION**=7, 7.0, 7.0.0 ou 7.0.0.0.

### 8 - Mode normal

L'application JMS utilise le mode normal du fournisseur de messagerie d'IBM MQ. Le mode normal utilise toutes les fonctions d'un gestionnaire de files d'attente IBM MQ pour implémenter JMS. Ce mode est optimisé pour utiliser l'API JMS 2.0 et ses fonctionnalités.

Si vous vous connectez à un gestionnaire de files d'attente avec un niveau de commande 800 ou supérieur, vous pouvez utiliser toutes les API et fonctions JMS 2.0 , telles que l'envoi asynchrone, la distribution différée ou l'abonnement partagé.

Si le gestionnaire de files d'attente spécifié dans les paramètres de fabrique de connexions n'est pas un gestionnaire de files d'attente IBM MQ 8.0.0 ou version ultérieure, la méthode `createConnection` échoue avec une exception `JMSFMQ0003`.

Le mode normal du fournisseur de messagerie IBM MQ utilise la fonction de partage de conversations et le nombre de conversations qui peuvent être partagées est contrôlé par la propriété **SHARECNV()** sur le canal de connexion du serveur. Si cette propriété a pour valeur 0, vous ne pouvez pas utiliser le mode normal du fournisseur de messagerie IBM MQ et la méthode `createConnection` échoue avec l'exception `JMSCC5007`.

## 7 - Mode normal avec restrictions

L'application JMS utilise le mode normal du fournisseur de messagerie IBM MQ avec des restrictions. Ce mode utilise l'API JMS 2.0, mais pas les nouvelles fonctions telles que les abonnements partagés, la distribution différée ou l'envoi asynchrone.

Si vous définissez **PROVIDERVERSION** sur 7, seul le fournisseur de messagerie IBM MQ normal avec le mode de restriction des opérations est disponible.

Si vous vous connectez en mode normal avec des restrictions à un gestionnaire de files d'attente avec un niveau de commande inférieur à 800, vous pouvez utiliser l'API JMS 2.0 , mais pas les fonctions d'envoi asynchrone, de distribution différée ou d'abonnement partagé.

Le mode normal du fournisseur de messagerie IBM MQ avec restrictions utilise la fonction de partage de conversations et le nombre de conversations qui peuvent être partagées est contrôlé par la propriété **SHARECNV()** sur le canal de connexion du serveur. Si cette propriété a pour valeur 0, vous ne pouvez pas utiliser le mode normal du fournisseur de messagerie IBM MQ avec restrictions et la méthode `createConnection` échoue avec l'exception `JMSCC5007`.

## 6 - Mode de migration

L'application JMS utilise le mode de migration du fournisseur de messagerie IBM MQ.

Vous pouvez vous connecter à un gestionnaire de files d'attente IBM MQ 8.0 ou version ultérieure à l'aide de ce mode, mais aucune des nouvelles fonctions d'un gestionnaire de files d'attente IBM MQ classes for JMS n'est utilisée, par exemple, la lecture anticipée ou la diffusion en flux.

Si vous disposez d'un client IBM MQ 8.0 ou ultérieur se connectant à un gestionnaire de files d'attente IBM MQ 8.0 ou ultérieur, la sélection des messages est effectuée par le gestionnaire de files d'attente plutôt que par le système client.

Si le mode de migration du fournisseur de messagerie IBM MQ est spécifié et que vous tentez d'utiliser l'une des API JMS 2.0, l'appel de méthode d'API échoue avec l'exception `JMSCC5007`.

### unspecified (valeur par défaut)

La propriété **PROVIDERVERSION** est définie sur *Non spécifié* par défaut.

Une fabrique de connexions qui a été créée avec une version précédente d'IBM MQ classes for JMS dans JNDI prend cette valeur lorsqu'elle est utilisée avec la nouvelle version d'IBM MQ classes for JMS. L'algorithme ci-dessous sert à déterminer le mode de fonctionnement utilisé. Cet algorithme est utilisé lorsque la méthode `createConnection` est appelée et utilise d'autres aspects de la fabrique de connexions afin de déterminer si le mode normal du fournisseur de messagerie IBM MQ, le mode normal avec restrictions ou le mode de migration du fournisseur de messagerie IBM MQ est requis.

1. Tout d'abord, une tentative d'utilisation du mode normal du fournisseur de messagerie IBM MQ est effectuée.
2. Si le gestionnaire de files d'attente connecté n'est pas un gestionnaire de files d'attente IBM MQ 8.0 ou version ultérieure, une tentative d'utilisation du mode normal avec restrictions du fournisseur de messagerie IBM MQ est effectuée.

3. Si le gestionnaire de files d'attente connecté n'est pas un gestionnaire de files d'attente IBM WebSphere MQ 7.0.1 ou version ultérieure, la connexion est fermée et le mode de migration du fournisseur de messagerie IBM MQ est utilisé à la place.
4. Si la propriété **SHARECNV** du canal de connexion du serveur est définie sur 0, la connexion est fermée et le mode de migration du fournisseur de messagerie IBM MQ est utilisé à la place.
5. Si **BROKERVER** est défini sur V1 ou sur la valeur par défaut *unspecified*, le mode normal du fournisseur de messagerie IBM MQ continue d'être utilisé.

Voir [ALTER QMGR](#) pour plus d'informations sur le paramètre PSMODE de la commande ALTER QMGR et la compatibilité.

6. Si **BROKERVER** est défini sur V2, l'action prise dépend de la valeur de **BROKERQMGR** :
  - Si **BROKERQMGR** est Vide :

Si la file d'attente spécifiée par la propriété **BROKERCONQ** peut être ouverte pour la sortie (c'est-à-dire que MQOPEN aboutit pour la sortie) et que **PSMODE** sur le gestionnaire de files d'attente a pour valeur COMPAT ou DISABLED, le mode de migration du fournisseur de messagerie d'IBM MQ est utilisé.
  - Si la file d'attente spécifiée par la propriété **BROKERCONQ** ne peut pas être ouverte pour la sortie, ou si l'attribut **PSMODE** est défini sur ENABLED :

Le mode normal de fournisseur de messagerie IBM MQ est utilisé.
  - Si **BROKERQMGR** est non vide :

Le mode de migration du fournisseur de messagerie IBM MQ est utilisé.

Si vous ne pouvez pas modifier la fabrique de connexions que vous utilisez, vous pouvez utiliser la propriété `com.ibm.msg.client.wmq.overrideProviderVersion` pour remplacer tout paramètre de la fabrique de connexions. Ce remplacement s'applique à toutes les fabriques de connexions de la machine virtuelle Java mais les objets de fabrique de connexions réels ne sont pas modifiés.

#### Tâches associées

Configuration de la propriété JMS **PROVIDERVERSION**

## PROXYHOSTNAME

Nom d'hôte ou adresse IP du système sur lequel le serveur proxy s'exécute lors de l'utilisation d'une connexion en temps réel à un courtier via un serveur proxy.

### Objets applicables

ConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS : PROXYHOSTNAME

Nom abrégé de l'outil d'administration JMS : PHOST

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setProxyHostName()
- MQConnectionFactory.getProxyHostName()

### Valeurs

**null**

Nom d'hôte du serveur proxy. Il s'agit de la valeur par défaut.

## PROXYPORT

Numéro du port sur lequel le serveur proxy est à l'écoute lors de l'utilisation d'une connexion en temps réel à un courtier via un serveur proxy.

### Objets applicables

ConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS : PROXYPORT

Nom abrégé de l'outil d'administration JMS : PPORT

### Accès par programme

Méthodes d'accès set / getters

MQConnectionFactory.setProxyPort ()

MQConnectionFactory.getProxyPort ()

### Valeurs

**443**

Numéro de port du serveur proxy. Il s'agit de la valeur par défaut.

## PUBACKINT

Nombre de messages publiés par un diffuseur de publications avant que IBM MQ classes for JMS ne demande un accusé de réception au courtier.

Lorsque vous réduisez la valeur de cette propriété, IBM MQ classes for JMS demande des accusés de réception plus souvent. Par conséquent, les performances de l'éditeur diminuent. Lorsque vous augmentez la valeur, IBM MQ classes for JMS prend plus de temps pour émettre une exception en cas d'échec du courtier. Cette propriété est pertinente uniquement si TRANSPORT a la valeur BIND ou CLIENT.

### Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : PROXYPORT

Nom abrégé de l'outil d'administration JMS : PPORT

### Accès par programme

Méthodes d'accès set / getters

MQConnectionFactory.setPubAckInterval()

MQConnectionFactory.getPubAckInterval()

### Valeurs

**25**

Tout entier positif peut être la valeur par défaut.

## PUTASYNCALOWED

Cette propriété détermine si les expéditeurs de message sont autorisés à utiliser les insertions asynchrones pour envoyer des messages à cette destination.

## Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS : PUTASYNCALLOWED

Nom abrégé de l'outil d'administration JMS : PAALD

## Accès par programme

Méthodes d'accès set / getters

MQDestination.setPutAsyncAllowed()

MQDestination.getPutAsyncAllowed()

## Valeurs

### AS\_DEST

Détermine si les insertions asynchrones sont autorisées conformément à la définition de rubrique ou de file d'attente. Il s'agit de la valeur par défaut.

### AS\_Q\_DEF

Détermine si les insertions asynchrones sont autorisées conformément à la définition de file d'attente.

### AS\_TOPIC\_DEF

Détermine si les insertions asynchrones sont autorisées conformément à la définition de rubrique.

### NO

Les insertions asynchrones ne sont pas autorisées.

### YES

Les insertions asynchrones sont autorisées.

## QMANAGER

Nom du gestionnaire de files d'attente auquel établir la connexion.

Toutefois, si votre application utilise une table de définition de canal du client pour se connecter à un gestionnaire de files d'attente, voir [Utilisation d'une table de définition de canal du client avec IBM MQ classes for JMS](#).

## Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, Queue, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : QMANAGER

Nom abrégé de l'outil d'administration JMS : QMGR

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.GestionnairesetQueue()
- MQConnectionFactory.GestionnairegetQueue()

## Valeurs

### "" (chaîne vide)

Toute chaîne peut être la valeur par défaut.

## QUEUE

Nom de la destination de file d'attente JMS . Correspond au nom de la file d'attente utilisée par le gestionnaire de files d'attente.

### Objets applicables

File d'attente

Nom long de l'outil d'administration JMS : QUEUE

Nom abrégé de l'outil d'administration JMS : QU

### Valeurs

#### N'importe quelle chaîne

Tout nom de file d'attente IBM MQ valide.

#### Référence associée

[Règles de dénomination des objets IBM MQ >](#)

## READAHEADALLOWED

Cette propriété détermine si les consommateurs de messages et les navigateurs de file d'attente sont autorisés à utiliser la lecture anticipée pour extraire des messages non persistants de cette destination dans une mémoire tampon interne avant de les recevoir.

### Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS : READAHEADALLOWED

Nom abrégé de l'outil d'administration JMS : RAALD

### Accès par programme

Méthodes d'accès set / getters

- MQDestination.setReadAheadAllowed()
- MQDestination.getReadAheadAllowed()

### Valeurs

#### AS\_DEST

Détermine si la lecture anticipée est autorisée conformément à la définition de file d'attente ou de rubrique. Il s'agit de la valeur par défaut dans les outils d'administration.

Utilisez WMQConstants.WMQ\_READ\_AHEAD\_ALLOWED\_AS\_DEST dans les programmes.

#### AS\_Q\_DEF

Détermine si la lecture anticipée est autorisée conformément à la définition de file d'attente.

Utilisez WMQConstants.WMQ\_READ\_AHEAD\_ALLOWED\_AS\_Q\_DEF dans les programmes.

#### AS\_TOPIC\_DEF

Détermine si la lecture anticipée est autorisée conformément à la définition de rubrique.

Utilisez WMQConstants.WMQ\_READ\_AHEAD\_ALLOWED\_AS\_TOPIC\_DEF dans les programmes.

#### NO

La lecture anticipée n'est pas autorisée.

Utilisez WMQConstants.WMQ\_READ\_AHEAD\_ALLOWED\_DISABLED dans les programmes.

## YES

La lecture anticipée est autorisée.

Utilisez `WMQConstants.WMQ_READ_AHEAD_ALLOWED_ENABLED` dans les programmes.

## READAHEADCLOSEPOLICY

Pour les messages distribués à un programme d'écoute de message asynchrone, qu'arrive-t-il aux messages de la mémoire tampon de lecture anticipée interne lorsque le consommateur de message est fermé?

### Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS : `READAHEADCLOSEPOLICY`

Nom abrégé de l'outil d'administration JMS : `RACP`

### Accès par programme

Méthodes d'accès set / getters

- `MQDestination.setReadAheadClosePolicy()`
- `MQDestination.getReadAheadClosePolicy()`

### Valeurs

#### DISTRIBUER

Tous les messages de la mémoire tampon de lecture anticipée interne sont distribués au programme d'écoute des messages de l'application avant d'être renvoyés. Il s'agit de la valeur par défaut dans les outils d'administration.

Utilisez `WMQConstants.WMQ_READ_AHEAD_DELIVERALL` dans les programmes.

#### EN COURS DE LIVRAISON

Seul l'appel du programme d'écoute des messages se termine avant d'être renvoyé, laissant potentiellement des messages dans la mémoire tampon interne de lecture anticipée qui sont ensuite supprimés.

Utilisez `WMQConstants.WMQ_READ_AHEAD_DELIVERCURRENT` dans les programmes.

## RECEIVECCSID

Propriété de destination qui définit le CCSID cible pour la conversion des messages du gestionnaire de files d'attente. La valeur est ignorée sauf si `RECEIVECONVERSION` est défini sur `WMQ_RECEIVE_CONVERSION_QMGR`

### Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS : `RECEIVECCSID`

Nom abrégé de l'outil d'administration JMS : `RCCS`

### Accès par programme

Méthodes d'accès set / Getters

- `MQDestination.setReceiveCCSID`
- `MQDestination.getReceiveCCSID`



## Valeurs

### **WMQConstants.WMQ\_RECEIVE\_CCsid\_JVM\_DEFAULT**

0 -Utiliser la machine virtuelle Java Charset.defaultCharset

### **1208**

UTF-8

### **ccsid**

Identificateur de jeu de caractères codés pris en charge.

## RECEIVECONVERSION

Propriété de destination qui détermine si la conversion de données sera effectuée par le gestionnaire de files d'attente.

### Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS : RECEIVECONVERSION

Nom abrégé de l'outil d'administration JMS : RCNV

### Accès par programme

#### Méthodes d'accès set / Getters

- `MQDestination.setReceiveConversion`
- `MQDestination.getReceiveConversion`

## Valeurs

### **WMQConstants.WMQ\_RECEIVE\_CONVERSION\_CLIENT\_MSG**

1 -Effectuer uniquement la conversion de données sur le client JMS . La valeur par défaut est comprise entre V7.0et 7.0.1.5.

### **WMQConstants.WMQ\_RECEIVE\_CONVERSION\_QMGR**

2 -Effectuer la conversion de données sur le gestionnaire de files d'attente avant d'envoyer un message au client. Valeur par défaut (et uniquement) comprise entre V7.0 et V7.0.1.4 inclus, sauf si l'APAR IC72897 est appliqué.

## RECEIVEISOLATION

Cette propriété détermine si un abonné peut recevoir des messages qui n'ont pas été validés dans la file d'attente de l'abonné.

### Objets applicables

ConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS : RECEIVEISOLATION

Nom abrégé de l'outil d'administration JMS : RCVISOL

## Valeurs

### **VALIDE**

Un abonné reçoit uniquement les messages de la file d'attente d'abonné qui ont été validés. Il s'agit de la valeur par défaut dans les outils d'administration.

Utilisez `WMQConstants.WMQ_RCVISOL_COMMITTED` dans les programmes.

## NON VALIDE

Un abonné peut recevoir des messages qui n'ont pas été validés dans la file d'attente de l'abonné.  
Utilisez `WMQConstants.WMQ_RCVISOL_UNCOMMITTED` dans les programmes.

## RECEXIT

Identifie un exit de réception de canal, ou une séquence d'exits de réception, à exécuter successivement.

Une configuration supplémentaire peut être nécessaire pour que le IBM MQ classes for JMS puisse localiser les exits de réception. Pour plus d'informations, voir [Configuration des classes IBM MQ pour JMS afin d'utiliser des exits de canal](#).

### Objets applicables

`ConnectionFactory`, `QueueConnectionFactory`, `TopicConnectionFactory`, `XAConnectionFactory`, `XAQueueConnectionFactory`, `XATopicConnectionFactory`

Nom long de l'outil d'administration JMS : RECEXIT

Nom abrégé de l'outil d'administration JMS : RCX

### Accès par programme

Méthodes d'accès set / getters

- `MQConnectionFactory.setReceiveExit ()`
- `MQConnectionFactory.getReceiveExit ()`

### Valeurs

- `Null`. Il s'agit de la valeur par défaut.
- Chaîne comprenant un ou plusieurs éléments séparés par des virgules, où chaque élément est:
  - Nom d'une classe qui implémente l'interface `WMQReceiveExit` (pour un exit de réception de canal écrit en Java).
  - Chaîne au format `libraryName(entryPointName)` (pour un exit de réception de canal non écrit en Java).

## RECEXITINIT

Données utilisateur transmises aux exits de réception de canal lorsqu'ils sont appelés.

### Objets applicables

`ConnectionFactory`, `QueueConnectionFactory`, `TopicConnectionFactory`, `XAConnectionFactory`, `XAQueueConnectionFactory`, `XATopicConnectionFactory`

Nom long de l'outil d'administration JMS : RECEXITINIT

Nom abrégé de l'outil d'administration JMS : RCXI

### Accès par programme

Méthodes d'accès set / getters

- `MQConnectionFactory.setReceiveExitInit()`
- `MQConnectionFactory.getReceiveExitInit()`

## Valeurs

### null

Chaîne comprenant un ou plusieurs éléments de données utilisateur séparés par des virgules. Il s'agit de la valeur par défaut.

## REPLYTOSTYLE

Détermine comment la zone JMSReplyTo d'un message reçu est construite.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : REPLYTOSTYLE

Nom abrégé de l'outil d'administration JMS : RTOST

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setReplyToStyle()
- MQConnectionFactory.getReplyToStyle()

## Valeurs

### DEFAULT

Equivalent à MQMD.

### RFH2

Utilisez la valeur fournie dans l'en-tête RFH2 . Si une valeur JMSReplyTo a été définie dans l'application émettrice, utilisez cette valeur.

### MQMD

Utilisez la valeur fournie par MQMD. Ce comportement est équivalent au comportement par défaut de IBM WebSphere MQ 6.0.2.4 et 6.0.2.5.

Si la valeur JMSReplyTo définie par l'application émettrice ne contient pas de nom de gestionnaire de files d'attente, le gestionnaire de files d'attente de réception insère son propre nom dans le MQMD. Si vous définissez ce paramètre sur MQMD, la file d'attente de réponse que vous utilisez se trouve sur le gestionnaire de files d'attente de réception. Si vous définissez ce paramètre sur RFH2, la file d'attente de réponse que vous utilisez se trouve sur le gestionnaire de files d'attente spécifié dans RFH2 du message envoyé, comme défini à l'origine par l'application émettrice.

Si la valeur JMSReplyTo définie par l'application émettrice contient un nom de gestionnaire de files d'attente, la valeur de ce paramètre n'est pas importante car MQMD et RFH2 contiennent la même valeur.

## RESCANINT

Lorsqu'un consommateur de message du domaine point à point utilise un sélecteur de message pour sélectionner les messages qu'il souhaite recevoir, IBM MQ classes for JMS recherche dans la file d'attente IBM MQ les messages appropriés dans la séquence déterminée par l'attribut MsgDeliverySequence de la file d'attente.

Une fois que IBM MQ classes for JMS a trouvé un message approprié et l'a envoyé au consommateur, IBM MQ classes for JMS reprend la recherche du message approprié suivant à partir de sa position actuelle dans la file d'attente. IBM MQ classes for JMS continue de rechercher la file d'attente de cette manière jusqu'à ce qu'elle atteigne la fin de la file d'attente ou jusqu'à ce que l'intervalle de temps en millisecondes, déterminé par la valeur de cette propriété, ait expiré. Dans chaque cas, IBM MQ classes for

JMS revient au début de la file d'attente pour poursuivre la recherche et un nouvel intervalle de temps commence.

## Objets applicables

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory

Nom long de l'outil d'administration JMS : RESCANINT

Nom abrégé de l'outil d'administration JMS : RINT

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setRescanIntervalle ()
- MQConnectionFactory.getRescanIntervalle ()

## Valeurs

**5000**

Tout entier positif peut être la valeur par défaut.

## SECEXIT

Identifie un exit de sécurité de canal.

Une configuration supplémentaire peut être nécessaire pour que le IBM MQ classes for JMS puisse localiser les exits de sécurité. Pour plus d'informations, voir [Configuration des classes IBM MQ pour JMS afin d'utiliser des exits de canal](#).

## Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : SECEXIT

Nom abrégé de l'outil d'administration JMS : SXC

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setSecurityExit ()
- MQConnectionFactory.getSecurityExit ()

## Valeurs

- Null. Il s'agit de la valeur par défaut.
- Chaîne comprenant un ou plusieurs éléments séparés par des virgules, où chaque élément est:
  - Nom d'une classe qui implémente l'interface WMQSecurityExit (pour un exit de sécurité de canal écrit en Java).
  - Chaîne au format *libraryName(entryPointName)* (pour un exit de sécurité de canal non écrit en Java).

## SECEXITINIT

Données utilisateur transmises à un exit de sécurité de canal lors de l'appel.

## Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : SECEXITINIT

Nom abrégé de l'outil d'administration JMS : SCXI

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setSecurityExitInit()
- MQConnectionFactory.getSecurityExitInit()

## Valeurs

null

Toute chaîne peut être la valeur par défaut.

## SENDCHECKCOUNT

Nombre d'appels d'envoi à autoriser entre deux vérifications d'erreurs d'insertion asynchrone, au sein d'une même session JMS non transactionnelle.

## Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : SENDCHECKCOUNT

Nom abrégé de l'outil d'administration JMS : SCC

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setSendCheckCount()
- MQConnectionFactory.getSendCheckCount()

## Valeurs

null

Toute chaîne peut être la valeur par défaut.

## SENDEXIT

Identifie un exit d'émission de canal ou une séquence d'exits d'émission à exécuter successivement.

Une configuration supplémentaire peut être nécessaire pour que le IBM MQ classes for JMS puisse localiser les exits d'émission. Pour plus d'informations, voir [Configuration des classes IBM MQ pour JMS](#) afin d'utiliser des exits de canal.

## Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : SENDEXIT

Nom abrégé de l'outil d'administration JMS : SDX

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setSendExit ()
- MQConnectionFactory.getSendExit ()

### Valeurs

- Null. Il s'agit de la valeur par défaut.
- Chaîne comprenant un ou plusieurs éléments séparés par des virgules, où chaque élément est:
  - Nom d'une classe qui implémente l'interface WMQSendExit (pour un exit d'émission de canal écrit en Java).
  - Chaîne au format *libraryName(entryPointNom)* (pour un exit d'émission de canal non écrit en Java).

## SENDEXITINIT

Données utilisateur qui sont transmises aux exits d'émission une fois ceux-ci appelés.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : SENDEXITINIT

Nom abrégé de l'outil d'administration JMS : SDXI

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setSendExitInit()
- MQConnectionFactory.getSendExitInit()

### Valeurs

null

Toute chaîne comprenant un ou plusieurs éléments de données utilisateur séparés par des virgules peut être la valeur par défaut.

## SHARECONVALLOWED

Pour les applications qui utilisent le mode normal du fournisseur de messagerie IBM MQ ou le mode normal avec restrictions, cette propriété détermine si la fonction de partage de conversations est utilisée pour les connexions, les sessions et les contextes JMS créés à partir de la fabrique de connexions.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : SHARECONVALLOWED

Nom abrégé de l'outil d'administration JMS : SCALD

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setShareConvAllowed()
- MQConnectionFactory.getShareConvAllowed()

## Valeurs

### YES

Les connexions, les sessions et les contextes JMS créés à partir de la fabrique de connexions dans la même machine virtuelle Java peuvent partager une instance de canal (qui est mappée à une connexion TCP/IP), le cas échéant.

Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez WMQConstants.WMQ\_SHARE\_CONV\_ALLOWED\_YES.

### NO

Chaque connexion JMS créée à partir de la fabrique de connexions et chaque session JMS créée à partir de ces connexions JMS possède sa propre instance de canal (connexion TCP/IP) vers un gestionnaire de files d'attente.

Pour les contextes JMS, le premier contexte créé à partir de la fabrique de connexions crée deux instances de canal (connexions TCP/IP). Les autres contextes JMS créés à partir du premier ont leur propre instance de canal (connexion TCP/IP).

Pour les programmes, utilisez WMQConstants.WMQ\_SHARE\_CONV\_ALLOWED\_NO.

## Concepts associés

[Modes de fonctionnement du fournisseur de messagerie IBM MQ](#)

[Partage d'une connexion TCP/IP dans IBM MQ classes for JMS](#)

## SPARSESUBS

Contrôle la stratégie d'extraction de message d'un objet TopicSubscriber .

## Objets applicables

ConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS : SPARSESUBS

Nom abrégé de l'outil d'administration JMS : SSUBS

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setSparseAbonnements ()
- MQConnectionFactory.getSparseAbonnements ()

## Valeurs

### NO

Les abonnements reçoivent des messages correspondants fréquents. Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez false.

### YES

Les abonnements reçoivent des messages correspondants peu fréquents. Cette valeur nécessite que la file d'attente d'abonnement puisse être ouverte pour l'exploration.

Pour les programmes, utilisez true.

## SSLCIPHERSUITE

CipherSuite à utiliser pour une connexion TLS.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : SSLCIPHERSUITE

Nom abrégé de l'outil d'administration JMS : SCPHS

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setSSLCipherSuite ()
- MQConnectionFactory.SuitegetSSLCipher()

### Valeurs

null

Il s'agit de la valeur par défaut. Pour plus d'informations, voir [Propriétés TLS des objets JMS](#).

## SSLCRL

Serveurs CRL pour la vérification de la révocation de certificat TLS.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : SSLCRL

Nom abrégé de l'outil d'administration JMS : SCRL

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setSSLCertMagasins ()
- MQConnectionFactory.getSSLCertMagasins ()

### Valeurs

null

Liste d'URL LDAP séparées par des espaces. Il s'agit de la valeur par défaut. Pour plus d'informations, voir [Propriétés TLS des objets JMS](#).

## SSLFIPSREQUIRED

Cette propriété détermine si une connexion TLS doit utiliser une CipherSuite prise en charge par le fournisseur IBM Java JSSE FIPS (IBMJSSEFIPS).

**Remarque :** Sous AIX, Linux, and Windows, IBM MQ fournit la conformité à la norme FIPS 140-2 via le module cryptographique IBM Crypto for C (ICC) . Le certificat de ce module a été déplacé vers le statut Historique. Les clients doivent afficher le certificat IBM Crypto for C (ICC) et prendre connaissance des conseils fournis par le NIST. Un module FIPS 140-3 de remplacement est actuellement en cours et son statut peut être affiché en le recherchant dans la [liste des modules NIST CMVP en cours de traitement](#).



## Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : SSLFIPSREQUIRED

Nom abrégé de l'outil d'administration JMS : SFIPS

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setSSLFipsObligatoire ()
- MQConnectionFactory.getSSLFipsObligatoire ()

## Valeurs

### NO

Une connexion TLS peut utiliser n'importe quelle CipherSuite qui n'est pas prise en charge par le fournisseur IBM Java JSSE FIPS (IBMJSSEFIPS).

Il s'agit de la valeur par défaut. Dans les programmes, utilisez false.

### YES

Une connexion TLS doit utiliser une CipherSuite prise en charge par IBMJSSEFIPS.

Dans les programmes, utilisez true.

## SSLPEERNAME

Pour TLS, squelette de *nom distinctif* qui doit correspondre à celui fourni par le gestionnaire de files d'attente.

## Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : SSLPEERNAME

Nom abrégé de l'outil d'administration JMS : SPEER

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setSSLPeerNom ()
- MQConnectionFactory.getSSLPeerNom ()

## Valeurs

### null

Il s'agit de la valeur par défaut. Pour plus d'informations, voir [Propriétés TLS des objets JMS](#).

## SSLRESETCOUNT

Pour TLS, nombre total d'octets envoyés et reçus par une connexion avant que la clé secrète utilisée pour le chiffrement ne soit renégociée.

## Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : SSLRESETCOUNT

Nom abrégé de l'outil d'administration JMS : SRC

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setSSLResetNombre ()
- MQConnectionFactory.getSSLResetNombre ()

## Valeurs

**0**

Zéro, ou tout entier positif inférieur ou égal à 999, 999, 999. Il s'agit de la valeur par défaut. Pour plus d'informations, voir [Propriétés TLS des objets JMS](#).

## STATREFRESHINT

Intervalle, en millisecondes, entre les actualisations de la transaction à exécution longue qui détecte lorsqu'un abonné perd sa connexion au gestionnaire de files d'attente.

Cette propriété est pertinente uniquement si la valeur de la substance est QUEUE.

## Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : STATREFRESHINT

Nom abrégé de l'outil d'administration JMS : SRI

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setStatusRefreshInterval()
- MQConnectionFactory.getStatusRefreshInterval()

## Valeurs

**60000**

Tout entier positif peut être la valeur par défaut. Pour plus d'informations, voir [Propriétés TLS des objets JMS](#).

## SUBSTORE

Où IBM MQ classes for JMS stocke les données persistantes relatives aux abonnements actifs.

## Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : CONTENU

Nom abrégé de l'outil d'administration JMS : SS

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setSubscriptionStore ()
- MQConnectionFactory.getSubscriptionStore ()

## Valeurs

### COURTIER

Utilisez le magasin d'abonnements basé sur un courtier pour conserver les détails des abonnements. Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez WMQConstants.WMQ\_SUBSTORE\_BROKER.

### MIGRATE

Transférez les informations d'abonnement du magasin d'abonnements basé sur la file d'attente vers le magasin d'abonnements basé sur le courtier.

Pour les programmes, utilisez WMQConstants.WMQ\_SUBSTORE\_MIGRATE.

### QUEUE

Utilisez le magasin d'abonnements basé sur une file d'attente pour conserver les détails des abonnements.

Pour les programmes, utilisez WMQConstants.WMQ\_SUBSTORE\_QUEUE.

## SYNCPOINTALLGETS

Cette propriété détermine si toutes les extractions doivent être effectuées sous le point de synchronisation.

## Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : SYNCPOINTALLGETS

Nom abrégé de l'outil d'administration JMS : SPAG

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setSyncpointAllGets()
- MQConnectionFactory.getSyncpointAllGets()

## Valeurs

### Non

Il s'agit de la valeur par défaut.

### Oui

## TARGCLIENT

Cette propriété détermine si le format IBM MQ RFH2 est utilisé pour échanger des informations avec des applications cible.

## Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS : TARGCLIENT

Nom abrégé de l'outil d'administration JMS : TC

### Accès par programme

Méthodes d'accès set / getters

- MQDestination.setTargetClient()
- MQDestination.getTargetClient()

### Valeurs

#### JMS

La cible du message est une application JMS . Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez WMQConstants.WMQ\_CLIENT\_JMS\_COMPLIANT.

#### MQ

La cible du message est une application nonJMS IBM MQ .

Pour les programmes, utilisez WMQConstants.WMQ\_CLIENT\_NONJMS\_MQ.

## TARGCLIENTMATCHING

Cette propriété détermine si un message de réponse, envoyé à la file d'attente identifiée par la zone d'en-tête JMSReplyTo d'un message entrant, a un en-tête MQRFH2 uniquement si le message entrant a un en-tête MQRFH2 .

### Objets applicables

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory

Nom long de l'outil d'administration JMS : TARGCLIENTMATCHING

Nom abrégé de l'outil d'administration JMS : TCM

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setTargetClientMatching()
- MQConnectionFactory.getTargetClientMatching()

### Valeurs

#### YES

Si un message entrant ne possède pas d'en-tête MQRFH2 , la propriété TARGCLIENT de l'objet Queue dérivée de la zone d'en-tête JMSReplyTo du message est envoyée à MQ. Si le message comporte un en-tête MQRFH2 , la propriété TARGCLIENT est définie sur JMS à la place. Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez true.

#### NO

La propriété TARGCLIENT de l'objet Queue dérivée de la zone d'en-tête JMSReplyTo d'un message entrant est toujours définie sur JMS.

Pour les programmes, utilisez false.

## TEMPMODEL

Nom de la file d'attente modèle à partir de laquelle les files d'attente temporaires JMS sont créées.

## Objets applicables

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory

Nom long de l'outil d'administration JMS : TEMPMODEL

Nom abrégé de l'outil d'administration JMS : TM

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setTemporaryModèle ()
- MQConnectionFactory.getTemporaryModèle ()

## Valeurs

### SYSTEM.DEFAULT.MODEL.QUEUE

Toute chaîne peut être la valeur par défaut.

## TEMPQPREFIX

Préfixe utilisé pour former le nom d'une file d'attente dynamique IBM MQ .

## Objets applicables

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory

Nom long de l'outil d'administration JMS : TEMPQPREFIX

Nom abrégé de l'outil d'administration JMS : TQP

## Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setTempQPrefix ()
- MQConnectionFactory.getTempQPrefix ()

## Valeurs

### " " (chaîne vide)

Le préfixe utilisé est CSQ.\* sur z/OS et AMQ.\* sur toutes les autres plateformes. Il s'agit des valeurs par défaut.

### préfixe de file d'attente

Le préfixe de file d'attente correspond à toute chaîne conforme aux règles de formation du contenu de la zone *DynamicQName* dans un descripteur d'objet IBM MQ (structure MQOD), mais le dernier caractère non vide doit être un astérisque.

## TEMPTOPICPREFIX

Lors de la création de rubriques temporaires, JMS génère une chaîne de rubrique au format " TEMP / TEMPTOPICPREFIX/unique\_id " ou, si cette propriété est laissée avec la valeur par défaut, uniquement " TEMP /unique\_id ". La spécification d'un TEMPTOPICPREFIX non vide permet de définir des files d'attente modèles spécifiques pour la création de files d'attente gérées pour les abonnés à des rubriques temporaires créées sous cette connexion.

## Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : TEMPTOPICPREFIX

Nom abrégé de l'outil d'administration JMS : TTP

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setTempTopicPrefix()
- MQConnectionFactory.getTempTopicPrefix()

### Valeurs

Toute chaîne non nulle constituée uniquement de caractères valides pour une chaîne de rubrique IBM MQ . La valeur par défaut est " " (chaîne vide).

## TOPIC

Nom de la destination de rubrique JMS . Cette valeur est utilisée par le gestionnaire de files d'attente en tant que chaîne de rubrique d'une publication ou d'un abonnement.

### Objets applicables

Topic

Nom long de l'outil d'administration JMS : TOPIC

Nom abrégé de l'outil d'administration JMS : TOP

### Valeurs

#### N'importe quelle chaîne

Chaîne qui forme une chaîne de rubrique IBM MQ valide. Lorsque vous utilisez IBM MQ comme fournisseur de messagerie avec WebSphere Application Server, indiquez une valeur correspondant au nom sous lequel la rubrique est connue dans WebSphere Application Server à des fins d'administration.

#### Référence associée

Chaînes de rubrique

## TRANSPORT

Nature d'une connexion à un gestionnaire de files d'attente ou à un courtier.

### Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS : TRANSPORT

Nom abrégé de l'outil d'administration JMS : TRAN

### Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setTransportType ()
- MQConnectionFactory.getTransportType ()

## Valeurs

### Berkeley Internet Name Domain

Pour une connexion à un gestionnaire de files d'attente en mode liaisons. Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez `WMQConstants.WMQ_CM_BINDINGS`.

### CLIENT

Pour une connexion à un gestionnaire de files d'attente en mode client.

Pour les programmes, utilisez `WMQConstants.WMQ_CM_CLIENT`.

### Directe

Pour une connexion en temps réel à un courtier qui n'utilise pas la tunnellation HTTP .

Pour les programmes, utilisez `WMQConstants.WMQ_CM_DIRECT_TCPIP`.

### DIRECTIVE THHTTP

Pour une connexion en temps réel à un courtier à l'aide de la tunnellation HTTP . Seul HTTP 1.0 est pris en charge.

Pour les programmes, utilisez `WMQConstants.WMQ_CM_DIRECT_HTTP`.

### Concepts associés

«Dépendances entre les propriétés des objets IBM MQ classes for JMS», à la page 1987  
La validité de certaines propriétés dépend des valeurs particulières d'autres propriétés.

## WILDCARDFORMAT

Cette propriété détermine la version de la syntaxe générique à utiliser.

### Objets applicables

`ConnectionFactory`, `TopicConnectionFactory`, `XAConnectionFactory`, `XATopicConnectionFactory`

Nom long de l'outil d'administration JMS : `WILDCARDFORMAT`

Nom abrégé de l'outil d'administration JMS : `WCFMT`

### Accès par programme

Méthodes d'accès set / getters

- `MQConnectionFactory.setWildcardFormat()`
- `MQConnectionFactory.getWildcardFormat()`

## Valeurs

### TOPIC\_ONLY

Reconnaît uniquement les caractères génériques de niveau rubrique, tels qu'ils sont utilisés dans la version 2 du courtier. Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez `WMQConstants.WMQ_WILDCARD_TOPIC_ONLY`.

### CHAR\_ONLY (UNIQUEMENT)

Reconnaît uniquement les caractères génériques, tels qu'ils sont utilisés dans la version 1 du courtier.

Pour les programmes, utilisez `WMQConstants.WMQ_WILDCARD_CHAR_ONLY`.

## Propriété ENCODING

La propriété `ENCODING` comprend trois sous-propriétés, en douze combinaisons possibles.

Les valeurs valides que la propriété `ENCODING` peut prendre sont construites à partir des trois sous-propriétés:

### Codage d'entier

Normale ou inversée

### Codage décimal

Normale ou inversée

### codage en virgule flottante

IEEE normal, IEEE inverse ou z/OS

La propriété ENCODING est exprimée sous la forme d'une chaîne de trois caractères avec la syntaxe suivante:

```
{N|R}{N|R}{N|R|3}
```

Dans cette chaîne:

- N indique normal
- R indique que la valeur est inversée
- 3 indique z/OS
- Le premier caractère représente le *codage d'entier*
- Le second caractère représente le *codage décimal*
- Le troisième caractère représente le *codage en virgule flottante*

Fournit un ensemble de douze valeurs possibles pour la propriété ENCODING .

Il existe une valeur supplémentaire, la chaîne NATIVE, qui définit les valeurs de codage appropriées pour la plateforme Java .

Les exemples suivants montrent des combinaisons valides pour ENCODING:

```
ENCODING (NNR)  
ENCODING (NATIVE)  
ENCODING (RR3)
```

## Propriétés TLS des objets JMS

Activez le chiffrement TLS (Transport Layer Security) à l'aide de la propriété SSLCIPHERSUITE. Vous pouvez ensuite modifier les caractéristiques du chiffrement TLS à l'aide de plusieurs autres propriétés.

Lorsque vous spécifiez TRANSPORT (CLIENT), vous pouvez activer la communication chiffrée TLS à l'aide de la propriété SSLCIPHERSUITE. Définissez cette propriété sur une CipherSuite valide fournie par votre fournisseur JSSE ; elle doit correspondre au CipherSpec nommé sur le canal SVRCONN nommé par la propriété CHANNEL.

Toutefois, les CipherSpecs (comme indiqué sur le canal SVRCONN) et les CipherSuites (comme indiqué sur les objets ConnectionFactory) utilisent des schémas de dénomination différents pour représenter les mêmes algorithmes de chiffrement TLS. Si un nom CipherSpec reconnu est spécifié dans la propriété SSLCIPHERSUITE, JMSAdmin émet un avertissement et mappe le CipherSpec à son équivalent CipherSuite. Voir [TLS CipherSpecs et CipherSuites dans IBM MQ classes for JMS](#) pour obtenir la liste des CipherSpecs reconnus par IBM MQ et JMSAdmin.

Si vous avez besoin d'une connexion pour utiliser une CipherSuite prise en charge par le fournisseur JSSE FIPS IBM Java (IBMJSSEFIPS), définissez la propriété SSLFIPSREQUIRED de la fabrique de connexions sur YES. La valeur par défaut de cette propriété est NO, ce qui signifie qu'une connexion peut utiliser n'importe quelle CipherSuite prise en charge. La propriété est ignorée si SSLCIPHERSUITE n'est pas défini.

Le paramètre SSLPEERNAME correspond au format du paramètre SSLPEER, qui peut être défini sur les définitions de canal. Il s'agit d'une liste de paires nom-valeur d'attribut séparées par des virgules ou des points-virgules. Exemple :



```
SSLPEERNAME(CN=QMGR.*, OU=IBM, OU=WEBSPHERE)
```

L'ensemble de noms et de valeurs constitue un *nom distinctif*. Pour plus de détails sur les noms distinctifs et leur utilisation avec IBM MQ, voir [Sécurisation de IBM MQ](#).

L'exemple donné vérifie le certificat d'identification présenté par le serveur lors de la connexion. Pour que la connexion aboutisse, le certificat doit avoir un nom usuel commençant par QMGR., et doit avoir au moins deux noms d'unité organisationnelle, le premier étant IBM et le second WEBSPHERE. La vérification n'est pas sensible à la casse.

Si SSLPEERNAME n'est pas défini, aucune vérification de ce type n'est effectuée. SSLPEERNAME est ignoré si SSLCIPHERSUITE n'est pas défini.

La propriété SSLCRL spécifie zéro ou plusieurs serveurs CRL (Certificate Revocation List). L'utilisation de cette propriété nécessite une machine virtuelle Java à l'adresse Java 2 v1.4. Il s'agit d'une liste délimitée par des espaces d'entrées au format suivant:

```
ldap:// hostname:[ port ]
```

éventuellement suivi d'un / unique. Si *port* est omis, le port LDAP par défaut 389 est utilisé. Lors de la connexion, le certificat TLS présenté par le serveur est vérifié par rapport aux serveurs CRL spécifiés. Pour plus d'informations sur la sécurité CRL, voir [Sécurisation de IBM MQ](#).

Si SSLCRL n'est pas défini, aucune vérification de ce type n'est effectuée. SSLCRL est ignoré si SSLCIPHERSUITE n'est pas défini.

La propriété SSLRESETCOUNT représente le nombre total d'octets envoyés et reçus par une connexion avant que la clé secrète utilisée pour le chiffrement ne soit renégociée. Le nombre d'octets envoyés est le nombre avant chiffrement et le nombre d'octets reçus est le nombre après déchiffrement. Le nombre d'octets inclut également les informations de contrôle envoyées et reçues par IBM MQ classes for JMS.

Par exemple, pour configurer un objet ConnectionFactory pouvant être utilisé pour créer une connexion via un canal MQI activé par TLS avec une clé secrète renégociée après la transmission de 4 Mo de données, exécutez la commande suivante à JMSAdmin:

```
ALTER CF(my.cf) SSLRESETCOUNT(4194304)
```

Si la valeur de SSLRESETCOUNT est zéro, qui est la valeur par défaut, la clé secrète n'est jamais renégociée. La propriété SSLRESETCOUNT est ignorée si SSLCIPHERSUITE n'est pas défini.

## Référence IBM MQ Message Service Client (XMS) for .NET

Cette section de référence fournit des informations sur les interfaces de classe IBM MQ Message Service Client (XMS) for .NET (XMS .NET) et sur les propriétés d'objet définies par XMS.

### .NET interfaces

Cette section décrit les interfaces de classe .NET ainsi que leurs propriétés et méthodes.

Le tableau suivant récapitule les interfaces qui sont définies dans l'espace de nom IBM.XMS.

Interface	Description
«IBytesMessage», à la page 2044	Un message d'octets est un message dont le corps contient un flux d'octets.
«IConnection», à la page 2053	Un objet Connection représente la connexion active de l'application à un serveur de messagerie.

Tableau 871. Récapitulatif des interfaces de classe .NET (suite)

Interface	Description
<a href="#">«IConnectionFactory»</a> , à la page 2056	Une application utilise une fabrique de connexions pour créer une connexion.
<a href="#">«IConnectionMetaData»</a> , à la page 2058	Un objet ConnectionMetaData fournit des informations sur une connexion.
<a href="#">«IDestination»</a> , à la page 2058	Une destination représente l'endroit où une application envoie des messages, l'endroit d'où elle en reçoit, ou les deux.
<a href="#">«ExceptionListener»</a> , à la page 2059	Une application utilise un programme d'écoute des exceptions afin d'être notifiée de manière asynchrone d'un problème lié à la connexion.
<a href="#">«IllegalStateException»</a> , à la page 2060	XMS émet cette exception si une application appelle une méthode à un moment incorrect ou si XMS n'est pas dans un état approprié pour l'opération demandée.
<a href="#">«InitialContext»</a> , à la page 2060	Une application utilise un objet InitialContext pour créer des objets à partir de définitions d'objet extraits d'un référentiel des objets gérés.
<a href="#">«InvalidClientIDException»</a> , à la page 2062	XMS émet cette exception si une application tente de définir un identificateur de client pour une connexion, mais que l'identificateur de client n'est pas valide ou qu'il est déjà utilisé.
<a href="#">«InvalidDestinationException»</a> , à la page 2062	XMS émet cette exception si une application spécifie une destination non valide.
<a href="#">«InvalidSelectorException»</a> , à la page 2063	XMS émet cette exception si une application fournit une expression de sélecteur de message dont la syntaxe n'est pas valide.
<a href="#">«IMapMessage»</a> , à la page 2063	Un message de mappe est un message dont le corps contient un ensemble de paires nom-valeur où un type de données est associé à chaque valeur.
<a href="#">«IMessage»</a> , à la page 2072	Un objet Message représente un message envoyé ou reçu par une application. IMessage est une superclasse des classes de messages comme IMapMessage.
<a href="#">«IMessageConsumer»</a> , à la page 2078	Une application utilise un consommateur de message pour la réception des messages envoyés vers une destination.
<a href="#">«MessageEOFException»</a> , à la page 2080	XMS émet cette exception si XMS rencontre la fin d'un flux de messages d'octets lorsqu'une application lit le corps d'un message d'octets.
<a href="#">«MessageFormatException»</a> , à la page 2080	XMS émet cette exception si XMS rencontre un message dont le format n'est pas valide.
<a href="#">«IMessageListener (délégation)»</a> , à la page 2081	Une application utilise un programme d'écoute de message pour recevoir des messages de manière asynchrone.

Tableau 871. Récapitulatif des interfaces de classe .NET (suite)

Interface	Description
<a href="#">«MessageNotReadableException»</a> , à la page 2081	XMS émet cette exception si une application tente de lire le corps d'un message en écriture seule.
<a href="#">«MessageNotWritableException»</a> , à la page 2081	XMS émet cette exception si une application tente d'écrire dans le corps d'un message en lecture seule.
<a href="#">«IMessageProducer»</a> , à la page 2082	Une application utilise un expéditeur de message pour envoyer des messages vers une destination.
<a href="#">«IObjectMessage»</a> , à la page 2087	Un message d'objet est un message dont le corps comprend un objet Java ou .NET sérialisé.
<a href="#">«IPropertyContext»</a> , à la page 2088	IPropertyContext est une superclasse abstraite qui contient des méthodes permettant d'obtenir et de définir des propriétés. Ces méthodes sont héritées par d'autres classes.
<a href="#">«IQueueBrowser»</a> , à la page 2097	Une application utilise un navigateur de files d'attente pour parcourir les messages sur une file d'attente sans les supprimer.
<a href="#">«Demandeur»</a> , à la page 2098	Une application utilise un demandeur pour envoyer un message de demande, puis attendre et recevoir la réponse.
<a href="#">«ResourceAllocationException»</a> , à la page 2100	XMS émet cette exception si XMS ne peut pas allouer les ressources requises par une méthode.
<a href="#">«SecurityException»</a> , à la page 2100	XMS émet cette exception si l'ID utilisateur et le mot de passe fournis pour authentifier une application sont rejetés. XMS émet également cette exception si une vérification des droits échoue et empêche l'aboutissement d'une méthode.
<a href="#">«ISession»</a> , à la page 2100	Une session est un contexte à unité d'exécution unique pour l'envoi et la réception de messages.
<a href="#">«IStreamMessage»</a> , à la page 2111	Un message de flux est un message dont le corps contient un flux de valeurs et où chaque valeur a un type de données associé.
<a href="#">«ITextMessage»</a> , à la page 2119	Un message texte est un message dont le corps contient une chaîne.
<a href="#">«TransactionInProgressException»</a> , à la page 2120	XMS émet cette exception si une application demande une opération qui n'est pas valide car une transaction est en cours.
<a href="#">«TransactionRolledBackException»</a> , à la page 2121	XMS émet cette exception si une application appelle Session.commit() pour valider la transaction en cours, mais que la transaction est ensuite annulée.
XMSC	Pour .NET, les noms et les valeurs de propriété XMS sont définis dans cette classe en tant que constantes publiques. Pour plus de détails, voir <a href="#">«Propriétés des objets XMS»</a> , à la page 2123.

Tableau 871. Récapitulatif des interfaces de classe .NET (suite)

Interface	Description
«XMSException», à la page 2121	<p>Si XMS détecte une erreur lors du traitement d'un appel à une méthode .NET , XMS émet une exception. Une exception est un objet qui encapsule des informations relatives à l'erreur.</p> <p>Il existe différents types d'exception XMS et un objet XMSException n'est qu'un seul type d'exception. Toutefois, la classe XMSException est une superclasse des classes d'exception XMS. XMS émet un objet XMSException si aucun autre type d'exception n'est approprié.</p>
«XMSFactoryFactory», à la page 2122	<p>Si une application n'utilise pas les objets gérés, utilisez cette classe pour créer des fabriques de connexions, des files d'attente et des rubriques.</p>

La définition de chaque méthode liste les codes d'exception que XMS peut retourner lorsqu'il détecte une erreur au cours du traitement d'un appel à la méthode. Chaque code d'exception est représenté par sa constante nommée, à laquelle correspond une exception.

## IBytesMessage

Un message d'octets est un message dont le corps contient un flux d'octets.

### Hierarchie d'héritage :

```

IBM.XMS.IPropertyContext
|
+---- IBM.XMS.IMessage
      |
      +---- IBM.XMS.IBytesMessage
    
```

## .NETpropriétés

*BodyLength* - Extraction de la longueur du corps

### Interface :

```

Int64 BodyLength
{
    get;
}
    
```

Extrait la longueur du corps du message en octets lorsque le corps du message est en lecture seule.

La valeur renvoyée correspond à la longueur de l'ensemble du corps, quelle que soit la position du curseur pour la lecture du message.

### Exceptions :

- XMSException
- MessageNotReadableException

## Méthodes

## *ReadBoolean - Lecture de valeur booléenne*

### **Interface :**

```
Boolean ReadBoolean();
```

Lit une valeur booléenne depuis le flux de message d'octets.

### **Paramètres :**

Aucun

### **Retour :**

Valeur booléenne lue.

### **Exceptions :**

- XMSEException
- MessageNotReadableException
- MessageEOFException

## *ReadSignedByte - Lecture d'octet*

### **Interface :**

```
Int16 ReadSignedByte();
```

Lit l'octet suivant du flux de message d'octets sous la forme d'un entier signé de 8 bits.

### **Paramètres :**

Aucun

### **Retour :**

Octet lu.

### **Exceptions :**

- XMSEException
- MessageNotReadableException
- MessageEOFException

## *ReadBytes - Lecture d'octets*

### **Interface :**

```
Int32 ReadBytes(Byte[] array);  
Int32 ReadBytes(Byte[] array, Int32 length);
```

Lit un tableau d'octets du flux de message d'octets, en commençant à la position du curseur.

### **Paramètres :**

#### **array (sortie)**

Mémoire tampon pour contenir le tableau d'octets lu. Si le nombre d'octets restant à lire à partir du flux avant l'appel est supérieur ou égal à la longueur de la mémoire tampon, celle-ci est remplie. Dans le cas contraire, elle est remplie partiellement avec les octets restants.

Si vous spécifiez un pointeur Null en entrée, la méthode ignore les octets sans les lire. Si le nombre d'octets restant à lire à partir du flux avant l'appel est supérieur ou égal à la longueur de la mémoire tampon, le nombre d'octets ignorés est égale à la longueur de la mémoire tampon. Sinon, les octets restants sont ignorés. Le curseur reste sur la position suivante pour lire le flux de message d'octets.

#### **length (entrée)**

Longueur de la mémoire tampon en octets

**Retour :**

Nombre d'octets lus dans la mémoire tampon. Si la mémoire tampon est partiellement remplie, la valeur est inférieure à la longueur de la mémoire tampon, indiquant qu'il ne reste plus d'octets à lire. S'il ne reste plus d'octets à lire à partir du flux avant l'appel, la valeur est XMSC\_END\_OF\_STREAM.

Si vous spécifiez un pointeur Null en entrée, la méthode ne renvoie aucune valeur.

**Exceptions :**

- XMSEException
- MessageNotReadableException

*ReadChar - Lecture de caractère*

**Interface :**

```
Char ReadChar();
```

Lit les 2 octets suivants à partir du flux de message d'octets sous la forme d'un caractère.

**Paramètres :**

Aucun

**Retour :**

Caractère lu.

**Exceptions :**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadDouble - Lecture d'un nombre en virgule flottante à double précision*

**Interface :**

```
Double ReadDouble();
```

Lit les 8 octets suivants à partir du flux de message d'octets sous la forme d'un nombre en virgule flottante à double précision.

**Paramètres :**

Aucun

**Retour :**

Le nombre en virgule flottante à double précision lu.

**Exceptions :**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadFloat - Lecture du nombre en virgule flottante*

**Interface :**

```
Single ReadFloat();
```

Lit les 4 octets suivants du flux de message d'octets sous la forme d'un nombre en virgule flottante.

**Paramètres :**

Aucun

**Retour :**

Le nombre en virgule flottante lu.

**Exceptions :**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadInt - Lecture d'un entier*

**Interface :**

```
Int32 ReadInt();
```

Lit les 4 octets suivants du flux de message d'octets sous la forme d'un entier signé de 32 bits.

**Paramètres :**

Aucun

**Retour :**

Entier lu.

**Exceptions :**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadLong - Lecture d'un entier long*

**Interface :**

```
Int64 ReadLong();
```

Lit les 8 octets suivants du flux de message d'octets sous la forme d'un entier signé de 64 bits.

**Paramètres :**

Aucun

**Retour :**

Entier long lu.

**Exceptions :**

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadShort - Lecture d'un entier court*

**Interface :**

```
Int16 ReadShort();
```

Lit les 2 octets suivants du flux de message d'octets sous la forme d'un entier signé de 16 bits.

**Paramètres :**

Aucun

**Retour :**

Entier court lu.

**Exceptions :**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadByte - Lecture d'octet non signé*

**Interface :**

```
Byte ReadByte();
```

Lit l'octet suivant du flux de message d'octets sous la forme d'un entier signé de 8 bits.

**Paramètres :**

Aucun

**Retour :**

Octet lu.

**Exceptions :**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadUnsignedShort - Lecture d'un entier court non signé*

**Interface :**

```
Int32 ReadUnsignedShort();
```

Lit les 2 octets suivants du flux de message d'octets sous la forme d'un entier non signé de 16 bits.

**Paramètres :**

Aucun

**Retour :**

Entier court non signé lu.

**Exceptions :**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadUTF - Lecture de chaîne UTF*

**Interface :**

```
String ReadUTF();
```

Lit une chaîne codée en UTF-8 à partir du flux de message d'octets.

**Remarque :** Avant d'appeler ReadUTF(), assurez-vous que le curseur de la mémoire tampon pointe sur le début du flux de message d'octets.

**Paramètres :**

Aucun

**Retour :**

Objet String encapsulant la chaîne lue.



**Exceptions :**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*Reset - Réinitialisation***Interface :**

```
void Reset();
```

Met le corps du message en mode lecture seule et repositionne le curseur au début du flux de message d'octets.

**Paramètres :**

Aucun

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageNotReadableException

*WriteBoolean - Ecriture d'une valeur booléenne***Interface :**

```
void WriteBoolean(Boolean value);
```

Ecrit une valeur booléenne sur le flux de message d'octets.

**Paramètres :****value (entrée)**

Valeur booléenne à écrire.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageNotWritableException

*WriteByte - Ecriture d'un octet***Interface :**

```
void WriteByte(Byte value);  
void WriteSignedByte(Int16 value);
```

Ecrit un octet sur le flux de message d'octets.

**Paramètres :****value (entrée)**

Octet à écrire.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageNotWritableException

*WriteBytes - Ecriture d'octets*

**Interface :**

```
void WriteBytes(Byte[] value);
```

Ecrit un tableau d'octets sur le flux de message d'octets.

**Paramètres :****value (entrée)**

Tableau des octets à écrire.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageNotWritableException

*WriteBytes - Ecriture d'un tableau partiel d'octets*

**Interface :**

```
void WriteBytes(Byte[] value, int offset, int length);
```

Ecrit un tableau partiel d'octets sur le flux de message d'octets, comme défini par la longueur spécifiée.

**Paramètres :****value (entrée)**

Tableau des octets à écrire.

**offset (entrée)**

Point de départ du tableau des octets à écrire.

**length (entrée)**

Nombre d'octets à écrire.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageNotWritableException

*WriteChar - Ecriture d'un caractère*

**Interface :**

```
void WriteChar(Char value);
```

Ecrit un caractère sur le flux de message d'octets sous forme de 2 octets, l'octet de poids fort en premier.

**Paramètres :****value (entrée)**

Caractère à écrire.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageNotWritableException

*WriteDouble - Ecriture d'un nombre en virgule flottante à double précision*

**Interface :**

```
void WriteDouble(Double value);
```

Convertit un nombre en virgule flottante à double précision en entier long et écrit celui-ci sur le flux de message d'octets sous forme de 8 octets, l'octet de poids fort en premier.

**Paramètres :****value (entrée)**

Le nombre en virgule flottante à double précision à écrire.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageNotWritableException

*WriteFloat - Ecriture d'un nombre en virgule flottante*

**Interface :**

```
void WriteFloat(Single value);
```

Convertit un nombre en virgule flottante en entier et écrit cet entier sur le flux de message d'octets sous forme de 4 octets, l'octet de poids fort en premier.

**Paramètres :****value (entrée)**

Le nombre en virgule flottante à écrire.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageNotWritableException

*WriteInt - Ecriture d'un entier*

**Interface :**

```
void WriteInt(Int32 value);
```

Écrit un entier sur le flux de message d'octets sous forme de 4 octets, l'octet de poids fort en premier.

**Paramètres :****value (entrée)**

Entier à écrire.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageNotWritableException

*WriteLong - Ecriture d'un entier long*

**Interface :**

```
void WriteLong(Int64 value);
```

Ecrit un entier long sur le flux de message d'octets sous forme de 8 octets, l'octet de poids fort en premier.

**Paramètres :****value (entrée)**

Entier long à écrire.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageNotWritableException

*WriteObject - Ecriture d'un objet*

**Interface :**

```
void WriteObject(Object value);
```

Ecrit l'objet spécifié dans le flux de message d'octets.

**Paramètres :****value (entrée)**

Objet à écrire, qui doit être une référence au type primitif.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageNotWritableException

*WriteShort - Ecriture d'un entier court*

**Interface :**

```
void WriteShort(Int16 value);
```

Ecrit un entier court sur le flux de message d'octets sous forme de 2 octets, l'octet de poids fort en premier.

**Paramètres :****value (entrée)**

Entier court à écrire.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageNotWritableException

*WriteUTF - Ecriture d'une chaîne UTF*

**Interface :**

```
void WriteUTF(String value);
```

Ecrit une chaîne codée en UTF-8 sur le flux de message d'octets.

**Paramètres :****value (entrée)**

Objet String encapsulant la chaîne à écrire.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageNotWritableException

**Propriétés et méthodes héritées**

Les propriétés suivantes sont héritées de l'interface [IMessage](#) :

[JMSCorrelationID](#), [JMSDeliveryMode](#), [JMSDestination](#), [JMSExpiration](#), [JMSMessageID](#), [JMSPriority](#), [JMSRedistribuée](#), [JMSReplyTo](#), [JMSTimestamp](#), [JMSType](#), [Propriétés](#)

Les méthodes suivantes sont héritées de l'interface [IMessage](#) :

[clearBody](#), [clearProperties](#), [PropertyExists](#)

Les méthodes suivantes sont héritées de l'interface [IPropertyContext](#) :

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

**IConnection**

Un objet Connection représente la connexion active de l'application à un serveur de messagerie.

**Hiérarchie d'héritage :**

```
IBM.XMS.IPropertyContext
|
+---- IBM.XMS.IConnection
```

Pour obtenir la liste des propriétés XMS définies d'un objet Connection, voir [«Propriétés de Connection»](#), à la page 2124.

**.NETpropriétés**

## ClientID - Extraction et définition de l'ID client

### Interface :

```
String ClientID
{
    get;
    set;
}
```

Obtient et définit l'identificateur de client pour la connexion.

L'identificateur de client peut être préconfiguré par l'administrateur dans une fabrique de connexions ou affectée par la définition de ClientID.

Un identificateur de client est utilisé uniquement pour prendre en charge des abonnements durables dans le domaine publication/abonnement ; il est ignoré dans le domaine point-à-point.

Si une application définit un identificateur de client pour une connexion, elle doit le faire immédiatement après la création de la connexion et avant toute autre opération sur cette connexion. Si l'application tente de définir un identificateur de client à un moment ultérieur, l'appel émet l'exception `IllegalStateException`.

Cette propriété n'est pas valide pour une connexion en temps réel à un courtier.

### Exceptions :

- `XMSEException`
- `IllegalStateException`
- `InvalidClientIDException`

## ExceptionListener - Extraction et définition d'un programme d'écoute des exceptions

### Interface :

```
ExceptionListener ExceptionListener
{
    get;
    set;
}
```

Extrait le programme d'écoute des exceptions enregistré avec la connexion et enregistre un programme d'écoute des exceptions avec la connexion.

Si aucun programme d'écoute des exceptions n'est enregistré avec la connexion, la méthode renvoie la valeur Null. Si un programme d'écoute des exceptions est déjà enregistré avec la connexion, vous pouvez annuler l'enregistrement en spécifiant une valeur Null au lieu du programme d'écoute des exceptions.

Pour plus d'informations sur l'utilisation des programmes d'écoute des exceptions, voir [Utilisation des programmes d'écoute des messages et des exceptions dans .NET](#).

### Exceptions :

- `XMSEException`

## Metadata - Extraction des métadonnées

### Interface :

```
IConnectionMetaData MetaData
{
    get;
}
```

Extrait les métadonnées pour la connexion.

**Exceptions :**

- XMSEException

**Méthodes**

*Close - Fermeture de la connexion*

**Interface :**

```
void Close();
```

Ferme la connexion.

Si une application tente de fermer une connexion déjà fermée, l'appel est ignoré.

**Paramètres :**

Aucun

**Retour :**

Vide

**Exceptions :**

- XMSEException

*CreateSession - Création de session*

**Interface :**

```
ISession CreateSession(Boolean transacted,  
                        AcknowledgeMode acknowledgeMode);
```

Crée une session.

**Paramètres :****transacted (entrée)**

La valeur `True` signifie que la session est transactionnelle. La valeur `False` signifie que la session n'est pas transactionnelle.

Pour une connexion en temps réel à un courtier, la valeur doit être `False`.

**acknowledgeMode (entrée)**

Indique la méthode d'accusé de réception des messages reçus par une application. L'énumérateur `AcknowledgeMode` doit utiliser l'une des valeurs suivantes :

```
AcknowledgeMode.AutoAcknowledge  
AcknowledgeMode.ClientAcknowledge  
AcknowledgeMode.DupsOkAcknowledge
```

Pour une connexion en temps réel à un courtier, la valeur doit être `AcknowledgeMode.AutoAcknowledge` ou `AcknowledgeMode.DupsOkAcknowledge`

Ce paramètre est ignoré si la session est transactionnelle. Pour plus d'informations sur les modes d'accusé de réception, voir [Accusé de réception de message](#).

**Retour :**

Objet `Session`

**Exceptions :**

- XMSEException

*Start - Démarrage d'une connexion*

**Interface :**

```
void Start();
```

Démarre ou redémarre la livraison des messages entrant pour la connexion. L'appel est ignoré si la connexion est déjà démarrée.

**Paramètres :**

Aucun

**Retour :**

Vide

**Exceptions :**

- XMSEException

*Stop - Arrêt de la connexion*

**Interface :**

```
void Stop();
```

Arrête la livraison des messages entrant pour la connexion. L'appel est ignoré si la connexion est déjà arrêtée.

**Paramètres :**

Aucun

**Retour :**

Vide

**Exceptions :**

- XMSEException

**Propriétés et méthodes héritées**

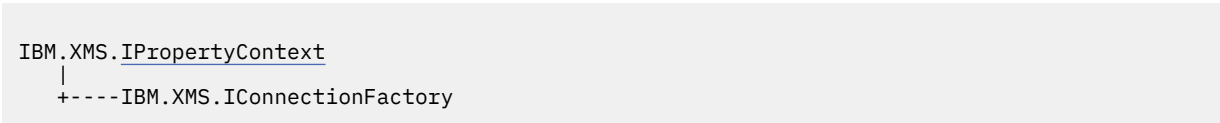
Les méthodes suivantes sont héritées de l'interface [IPropertyContext](#) :

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

**IConnectionFactory**

Une application utilise une fabrique de connexions pour créer une connexion.

**Hiérarchie d'héritage :**



Pour obtenir la liste des propriétés XMS définies d'un objet ConnectionFactory, voir «[Propriétés de ConnectionFactory](#)», à la page 2125.

**Méthodes**



*CreateConnection - Création d'une fabrique de connexions (avec ID utilisateur par défaut)*

**Interface :**

```
IConnection CreateConnection();
```

Crée une fabrique de connexions avec les propriétés par défaut.

Si vous vous connectez à IBM MQ alors que XMSC\_USERID n'est pas défini, le gestionnaire de files d'attente utilise l'ID utilisateur de l'utilisateur connecté par défaut. Si vous avez besoin d'authentifications de niveau connexion pour des utilisateurs individuels, vous pouvez écrire un exit d'authentification de client, configuré dans IBM MQ.

**Paramètres :**

Aucun

**Exceptions :**

- XMSEException

*CreateConnection - Création d'une connexion (avec ID utilisateur spécifié)*

**Interface :**

```
IConnection CreateConnection(String userId, String password);
```

Crée une connexion avec une identité d'utilisateur spécifiée.

Si vous vous connectez à IBM MQ alors que XMSC\_USERID n'est pas défini, le gestionnaire de files d'attente utilise l'ID utilisateur de l'utilisateur connecté par défaut. Si vous avez besoin d'authentifications de niveau connexion pour des utilisateurs individuels, vous pouvez écrire un exit d'authentification de client, configuré dans IBM MQ.

La connexion est créée en mode arrêté. Aucun message n'est distribué jusqu'à ce que l'application appelle **Connection.start()**.

**Paramètres :**

**userID (entrée)**

Objet String encapsulant l'ID utilisateur à utiliser pour authentifier l'application. Si vous indiquez une valeur Null, une tentative de création de connexion sans authentification est effectuée.

**password (entrée)**

Objet String encapsulant le mot de passe à utiliser pour authentifier l'application. Si vous indiquez une valeur Null, une tentative de création de connexion sans authentification est effectuée.

**Retour :**

Objet Connection.

**Exceptions :**

- XMSEException
- XMS\_X\_SECURITY\_EXCEPTION

**Propriétés et méthodes héritées**

Les méthodes suivantes sont héritées de l'interface [IPropertyContext](#) :

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

## IConnectionMetaData

Un objet ConnectionMetaData fournit des informations sur une connexion.

### Hiérarchie d'héritage :

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IConnectionMetaData
```

Pour obtenir la liste des propriétés XMS définies d'un objet ConnectionMetaData, voir [«Propriétés de ConnectionMetaData»](#), à la page 2129.

### .NETpropriétés

*JMSXPropertyNames* - Extraction des propriétés de message JMS définies

#### Interface :

```
System.Collections.IEnumerator JMSXPropertyNames
{
    get;
}
```

Renvoie une énumération des noms des propriétés de message définies par JMS et prises en charge par la connexion.

Les propriétés de message JMS définis ne sont pas prises en charge par une connexion en temps réel à un courtier.

#### Exceptions :

- XMSException

### Propriétés et méthodes héritées

Les méthodes suivantes sont héritées de l'interface [IPropertyContext](#) :

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

## IDestination

Une destination représente l'endroit où une application envoie des messages, l'endroit d'où elle en reçoit, ou les deux.

### Hiérarchie d'héritage :

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IDestination
```

Pour obtenir la liste des propriétés XMS définies d'un objet Destination, voir [«Propriétés de Destination»](#), à la page 2130.

### .NETpropriétés

*Name - Extraction du nom de destination*

**Interface :**

```
String Name
{
    get;
}
```

Extrait le nom de la destination. Le nom est une chaîne encapsulant le nom d'une file d'attente ou le nom d'une rubrique.

**Exceptions :**

- XMSEException

*TypeId - Extraction du type de destination*

**Interface :**

```
DestinationType TypeId
{
    get;
}
```

Extrait le type de la destination. Le type de la destination peut être l'une des valeurs suivantes :

DestinationType.Queue  
DestinationType.Topic

**Exceptions :**

- XMSEException

**Propriétés et méthodes héritées**

Les méthodes suivantes sont héritées de l'interface [IPropertyContext](#) :

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

**ExceptionListener**

Une application utilise un programme d'écoute des exceptions afin d'être notifiée de manière asynchrone d'un problème lié à la connexion.

**Hiérarchie d'héritage :**

Aucun

Si une application utilise une connexion uniquement pour consommer des messages de manière asynchrone, le programme d'écoute des exceptions constitue le seul moyen pour cette application d'être prévenue des problèmes liés à la connexion. Dans les autres cas, un programme d'écoute des exceptions peut fournir une manière plus directe de prendre connaissance d'un problème de connexion que d'attendre l'appel synchrone suivant à XMS.

**Délégation**

## ExceptionHandler – Programme d'écoute des exceptions

### Interface :

```
public delegate void ExceptionListener(Exception ex)
```

Notifie l'application d'un problème avec une connexion.

Les méthodes qui implémentent cette délégation peuvent être enregistrées avec la connexion.

Pour plus d'informations sur l'utilisation des programmes d'écoute des exceptions, voir [Utilisation des programmes d'écoute des messages et des exceptions dans .NET](#).

### Paramètres :

#### exception (entrée)

Pointeur vers une exception créée par XMS.

### Retour :

Vide

## IllegalStateException

XMS émet cette exception si une application appelle une méthode à un moment incorrect ou si XMS n'est pas dans un état approprié pour l'opération demandée.

### Hierarchie d'héritage :

```
IBM.XMS.XMSException
|
+----IBM.XMS.Exception
|
+----IBM.XMS.IllegalStateException
```

### Propriétés et méthodes héritées

Les méthodes suivantes sont héritées de l'interface [XMSException](#) :

[GetErrorCode](#), [GetLinkedException](#)

## InitialContext

Une application utilise un objet InitialContext pour créer des objets à partir de définitions d'objet extraits d'un référentiel des objets gérés.

### Hierarchie d'héritage :

Aucun

## .NETpropriétés

*Environment - Extraction de l'environnement*

### Interface :

```
Hashtable Environment
{
    get;
}
```

Extrait l'environnement.

### Exceptions :

- Les exceptions sont spécifiques au service d'annuaire en cours d'utilisation.

## Constructeurs

*InitialContext – Création d'un contexte initial*

### Interface :

```
InitialContext(Hashtable env);
```

Crée un objet InitialContext.

### Paramètres :

Les informations requises pour établir une connexion au référentiel des objets gérés est fourni pour le constructeur dans un environnement Hashtable.

### Exceptions :

- XMSEException

## Méthodes

*AddToEnvironment - Ajout d'une propriété à l'environnement*

### Interface :

```
Object AddToEnvironment(String propName, Object propVal);
```

Ajout d'une propriété à l'environnement.

### Paramètres :

#### **propName (entrée)**

Objet de type String encapsulant le nom de la propriété à ajouter.

#### **propVal (entrée)**

Valeur de la propriété à ajouter.

### Retour :

Ancienne valeur de la propriété.

### Exceptions :

- Les exceptions sont spécifiques au service d'annuaire en cours d'utilisation.

*Close - Fermeture de ce contexte*

### Interface :

```
void Close()
```

Ferme ce contexte.

### Paramètres :

Aucun

### Retour :

Aucun

### Exceptions :

- Les exceptions sont spécifiques au service d'annuaire en cours d'utilisation.

*Lookup – Recherche d'un objet dans un contexte initial*

**Interface :**

```
Object Lookup(String name);
```

Crée un objet à partir d'une définition d'objet extraite du référentiel des objets gérés.

**Paramètres :**

**name (entrée)**

Objet de type String encapsulant le nom de l'objet géré à extraire. Il peut s'agir d'un nom simple ou complexe. Pour plus de détails, voir [Extraction d'objets gérés](#).

**Retour :**

Objet IConnectionFactory ou IDestination, selon le type d'objet extrait. Si la fonction peut accéder au répertoire mais ne trouve pas l'objet requis, une valeur Null est renvoyée.

**Exceptions :**

- Les exceptions sont spécifiques au service d'annuaire en cours d'utilisation.

*RemoveFromEnvironment - Suppression d'une propriété de l'environnement*

**Interface :**

```
Object RemoveFromEnvironment(String propName);
```

Supprime une propriété de l'environnement.

**Paramètres :**

**propName (entrée)**

Objet de type String encapsulant le nom de la propriété à supprimer.

**Retour :**

Objet ayant été supprimé.

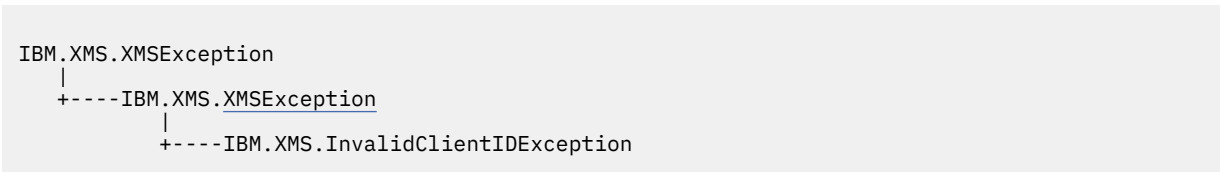
**Exceptions :**

- Les exceptions sont spécifiques au service d'annuaire en cours d'utilisation.

### **InvalidClientIDException**

XMS émet cette exception si une application tente de définir un identificateur de client pour une connexion, mais que l'identificateur de client n'est pas valide ou qu'il est déjà utilisé.

**Hiérarchie d'héritage :**



### **Propriétés et méthodes héritées**

Les méthodes suivantes sont héritées de l'interface [XMSEException](#) :

[GetErrorCode](#), [GetLinkedException](#)

### **InvalidDestinationException**

XMS émet cette exception si une application spécifie une destination non valide.

## Hiérarchie d'héritage :

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
      |
      +----IBM.XMS.InvalidDestinationException
```

### Propriétés et méthodes héritées

Les méthodes suivantes sont héritées de l'interface [XMSEException](#) :

[GetErrorCode](#), [GetLinkedException](#)

## InvalidSelectorException

XMS émet cette exception si une application fournit une expression de sélecteur de message dont la syntaxe n'est pas valide.

## Hiérarchie d'héritage :

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
      |
      +----IBM.XMS.InvalidSelectorException
```

### Propriétés et méthodes héritées

Les méthodes suivantes sont héritées de l'interface [XMSEException](#) :

[GetErrorCode](#), [GetLinkedException](#)

## IMapMessage

Un message de mappe est un message dont le corps contient un ensemble de paires nom-valeur où un type de données est associé à chaque valeur.

## Hiérarchie d'héritage :

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IMessage
      |
      +----IBM.XMS.IMapMessage
```

Lorsqu'une application obtient la valeur d'une paire nom-valeur, cette valeur peut être convertie par XMS en un autre type de données. Pour plus d'informations sur cette forme de conversion implicite, voir les informations sur les messages de mappe dans [Corps d'un message XMS](#).

## .NETpropriétés

*MapNames* - Extraction de noms de mappe

### Interface :

```
System.Collections.IEnumerator MapNames
{
    get;
}
```

Extrait une énumération des noms présents dans le corps du message de mappe.

**Exceptions :**

- XMSEException

**Méthodes**

*GetBoolean - Extraction d'une valeur booléenne*

**Interface :**

```
Boolean GetBoolean(String name);
```

Extrait la valeur booléenne identifiée par nom à partir du corps du message de mappe.

**Paramètres :****name (entrée)**

un objet String encapsulant le nom qui identifie la valeur booléenne.

**Retour :**

La valeur booléenne extraite du corps du message de mappe.

**Exceptions :**

- XMSEException

*GetByte - Extraction d'un octet*

**Interface :**

```
Byte GetByte(String name);  
Int16 GetSignedByte(String name);
```

Extraction de l'octet identifié par nom à partir du corps du message de mappe.

**Paramètres :****name (entrée)**

Un objet String encapsulant le nom qui identifie l'octet.

**Retour :**

Octet extrait à partir du corps du message de mappe. Aucune conversion de données n'est effectuée sur l'octet.

**Exceptions :**

- XMSEException

*GetBytes - Extraction d'octets*

**Interface :**

```
Byte[] GetBytes(String name);
```

Extrait le tableau d'octets identifié par nom à partir du corps du message de mappe.

**Paramètres :****name (entrée)**

Objet String encapsulant le nom qui identifie le tableau d'octets.

**Retour :**

Nombre d'octets dans le tableau.

**Exceptions :**

- XMSEException



*GetChar - Extraction d'un caractère*

**Interface :**

```
Char GetChar(String name);
```

Extrait le caractère identifié par nom à partir du corps du message de mappe.

**Paramètres :**

**name (entrée)**

Objet String encapsulant le nom qui identifie le caractère.

**Retour :**

Caractère extrait à partir du corps du message de mappe.

**Exceptions :**

- XMSEException

*GetDouble - Extraction du nombre en virgule flottante à double précision*

**Interface :**

```
Double GetDouble(String name);
```

Extrait le nombre en virgule flottante à double précision identifié par nom à partir du corps du message de mappe.

**Paramètres :**

**name (entrée)**

Objet String encapsulant le nom qui identifie le nombre en virgule flottante à double précision.

**Retour :**

Nombre en virgule flottante à double précision extrait du corps du message de mappe.

**Exceptions :**

- XMSEException

*GetFloat - Extraction d'un nombre en virgule flottante*

**Interface :**

```
Single GetFloat(String name);
```

Extrait le nombre en virgule flottante identifié par nom à partir du corps du message de mappe.

**Paramètres :**

**name (entrée)**

Objet String encapsulant le nom qui identifie le nombre en virgule flottante.

**Retour :**

Nombre en virgule flottante extrait du corps du message de mappe.

**Exceptions :**

- XMSEException

*GetInt - Extraction d'un entier*

**Interface :**

```
Int32 GetInt(String name);
```

Extrait l'entier identifié par nom à partir du corps du message de mappe.

**Paramètres :**

**name (entrée)**

Objet String encapsulant le nom qui identifie l'entier.

**Retour :**

Entier extrait à partir du corps du message de mappe.

**Exceptions :**

- XMSEException

*GetLong - Extraction d'un entier long*

**Interface :**

```
Int64 GetLong(String name);
```

Extrait l'entier long identifié par nom à partir du corps du message de mappe.

**Paramètres :**

**name (entrée)**

Objet String encapsulant le nom qui identifie l'entier long.

**Retour :**

Entier long extrait à partir du corps du message de mappe.

**Exceptions :**

- XMSEException

*GetObject - Extraction d'un objet*

**Interface :**

```
Object GetObject(String name);
```

Extrait une référence à la valeur d'une paire nom-valeur, à partir du corps du message de mappe. La paire nom-valeur est identifiée par nom.

**Paramètres :**

**name (entrée)**

Objet String encapsulant le nom de la paire nom-valeur.

**Retour :**

La valeur, dont le type d'objet peut être :

Boolean  
Byte  
Byte[]  
Char  
Double  
Single  
Int32  
Int64  
Int16  
String

**Exceptions :**

- XMSEException

*GetShort - Extrait un entier court*

**Interface :**

```
Int16 GetShort(String name);
```

Extrait l'entier court identifié par nom à partir du corps du message de mappe.

**Paramètres :**

**name (entrée)**

Objet String encapsulant le nom qui identifie l'entier court.

**Retour :**

Entier court extrait du corps du message de mappe.

**Exceptions :**

- XMSEException

*GetString - Extraction d'une chaîne*

**Interface :**

```
String GetString(String name);
```

Extrait la chaîne identifiée par nom à partir du corps du message de mappe.

**Paramètres :**

**name (entrée)**

Objet String encapsulant le nom qui identifie la chaîne dans le corps du message de mappe.

**Retour :**

Objet String encapsulant la chaîne extraite du corps du message de mappe. Si une conversion de données est requise, cette valeur correspond à la chaîne après conversion.

**Exceptions :**

- XMSEException

*ItemExists - Vérification de l'existence d'une paire nom-valeur*

**Interface :**

```
Boolean ItemExists(String name);
```

Vérifie si le corps du message de mappe contient une paire nom-valeur du nom spécifié.

**Paramètres :**

**name (entrée)**

Objet String encapsulant le nom de la paire nom-valeur.

**Retour :**

- True si le corps du message de mappe contient une paire nom-valeur du nom spécifié.
- False si le corps du message de mappe ne contient pas de paire nom-valeur du nom spécifié.

**Exceptions :**

- XMSEException

*SetBoolean - Définition d'une valeur booléenne*

**Interface :**

```
void SetBoolean(String name, Boolean value);
```

Définit une valeur booléenne dans le corps du message de mappe.

**Paramètres :****name (entrée)**

Objet String encapsulant le nom qui identifie la valeur booléenne dans le corps du message de mappe.

**value (entrée)**

Valeur booléenne à définir.

**Retour :**

Vide

**Exceptions :**

- XMSEException

*SetByte - Définition d'un octet*

**Interface :**

```
void SetByte(String name, Byte value);  
void SetSignedByte(String name, Int16 value);
```

Définit un octet dans le corps du message de mappe.

**Paramètres :****name (entrée)**

Objet String encapsulant le nom identifiant l'octet dans le corps du message de mappe.

**value (entrée)**

Octet à définir.

**Retour :**

Vide

**Exceptions :**

- XMSEException

*SetBytes - Définition d'octets*

**Interface :**

```
void SetBytes(String name, Byte[] value);
```

Définit un tableau d'octets dans le corps du message de mappe.

**Paramètres :****name (entrée)**

Objet String encapsulant le nom identifiant le tableau d'octets dans le corps du message de mappe.

**value (entrée)**

Tableau d'octets à définir.

**Retour :**

Vide

**Exceptions :**

- XMSEException

## *SetChar - Définition d'un caractère*

### **Interface :**

```
void SetChar(String name, Char value);
```

Définit un caractère sur 2 octets dans le corps du message de mappe.

### **Paramètres :**

#### **name (entrée)**

Objet String encapsulant le nom identifiant le caractère dans le corps du message de mappe.

#### **value (entrée)**

Caractère à définir.

### **Retour :**

Vide

### **Exceptions :**

- XMSEException

## *SetDouble - Définition du nombre en virgule flottante à double précision*

### **Interface :**

```
void SetDouble(String name, Double value);
```

Définit un nombre en virgule flottante à double précision dans le corps du message de mappe.

### **Paramètres :**

#### **name (entrée)**

Objet String encapsulant le nom identifiant le nombre en virgule flottante à double précision dans le corps du message de mappe.

#### **value (entrée)**

Le nombre en virgule flottante à double précision à définir.

### **Retour :**

Vide

### **Exceptions :**

- XMSEException

## *SetFloat - Définition d'un nombre en virgule flottante*

### **Interface :**

```
void SetFloat(String name, Single value);
```

Définit un nombre en virgule flottante dans le corps du message de mappe.

### **Paramètres :**

#### **name (entrée)**

Objet String encapsulant le nom identifiant le nombre en virgule flottante dans le corps du message de mappe.

#### **value (entrée)**

Nombre en virgule flottante à définir.

### **Retour :**

Vide

**Exceptions :**

- XMSEException

*SetInt - Définition d'un entier*

**Interface :**

```
void SetInt(String name, Int32 value);
```

Définit un entier dans le corps du message de mappe.

**Paramètres :****name (entrée)**

Objet String encapsulant le nom identifiant l'entier dans le corps du message de mappe.

**value (entrée)**

Entier à définir.

**Retour :**

Vide

**Exceptions :**

- XMSEException

*SetLong - Définition d'un entier long*

**Interface :**

```
void SetLong(String name, Int64 value);
```

Définit un entier long dans le corps du message de mappe.

**Paramètres :****name (entrée)**

Objet String encapsulant le nom identifiant l'entier long dans le corps du message de mappe.

**value (entrée)**

Entier long à définir.

**Retour :**

Vide

**Exceptions :**

- XMSEException

*SetObject - Définition d'un objet*

**Interface :**

```
void SetObject(String name, Object value);
```

Définit une valeur, qui doit être un type primitif XMS, dans le corps du message de mappe.

**Paramètres :****name (entrée)**

Objet String encapsulant le nom identifiant la valeur dans le corps du message de mappe.

**value (entrée)**

Tableau d'octets contenant la valeur à définir.

**Retour :**

Vide

**Exceptions :**

- XMSEException

*SetShort - Définition d'un entier court*

**Interface :**

```
void SetShort(String name, Int16 value);
```

Définit un entier court dans le corps du message de mappe.

**Paramètres :****name (entrée)**

Objet String encapsulant le nom identifiant l'entier court dans le corps du message de mappe.

**value (entrée)**

Entier court à définir.

**Retour :**

Vide

**Exceptions :**

- XMSEException

*SetString - Définition d'une chaîne*

**Interface :**

```
void SetString(String name, String value);
```

Définit une chaîne dans le corps du message de mappe.

**Paramètres :****name (entrée)**

Objet String encapsulant le nom identifiant la chaîne dans le corps du message de mappe.

**value (entrée)**

Objet de type String encapsulant la chaîne à définir.

**Retour :**

Vide

**Exceptions :**

- XMSEException

**Propriétés et méthodes héritées**

Les propriétés suivantes sont héritées de l'interface IMessage :

JMSCorrelationID, JMSDeliveryMode, JMSDestination, JMSExpiration, JMSMessageID, JMSPriority, JMSRedistribuée, JMSReplyTo, JMSTimestamp, JMSType, Propriétés

Les méthodes suivantes sont héritées de l'interface IMessage :

clearBody, clearProperties, PropertyExists

Les méthodes suivantes sont héritées de l'interface IPropertyContext :

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

## IMessage

Un objet Message représente un message envoyé ou reçu par une application. IMessage est une superclasse des classes de messages comme IMapMessage.

### Hiérarchie d'héritage :

```
IBM.XMS.IPropertyContext
|
+---- IBM.XMS.IMessage
```

Pour obtenir la liste des zones d'en-tête de message JMS dans un objet Message, voir [Zones d'en-tête d'un message XMS](#). Pour obtenir la liste des propriétés définies par JMS d'un objet Message, voir [Propriétés définies par JMS d'un message](#). Pour obtenir la liste des propriétés définies par IBM d'un objet Message, voir [IBM d'un message](#). Pour obtenir la liste des propriétés JMS\_IBM\_MQMD\* de l'objet Message, voir [«Propriétés JMS\\_IBM\\_MQMD\\*»](#), à la page 2134

Les messages sont supprimés par le récupérateur de place. Lorsqu'un message est supprimé, les ressources qu'il utilisait sont libérées.

### .NETpropriétés

*GetJMSCorrelationID - Extraction et définition de JMSCorrelationID*

#### Interface :

```
String JMSCorrelationID
{
    get;
    set;
}
```

Extrait et définit l'identificateur de corrélation du message en tant qu'objet String.

#### Exceptions :

- XMSException

*JMSDeliveryMode - Extraction et définition de JMSDeliveryMode*

#### Interface :

```
DeliveryMode JMSDeliveryMode
{
    get;
    set;
}
```

Extraction et définition du mode de livraison du message.

Le mode de livraison du message peut prendre l'une des valeurs suivantes :

```
DeliveryMode.Persistent
DeliveryMode.NonPersistent
```

Pour un nouveau message qui n'a pas été envoyé, le mode de livraison est `DeliveryMode.Persistent`, sauf dans le cas d'une connexion en temps réel à un courtier pour lequel le mode de livraison est `DeliveryMode.NonPersistent`. Pour un message reçu, la méthode renvoie le mode de livraison qui a été défini par l'appel `IMessageProducer.send()` lorsque le message a été envoyé, sauf si l'application de réception change le mode de livraison avec `JMSDeliveryMode`.

#### Exceptions :

- XMSException



## *JMSDestination - Extraction et définition de JMSDestination*

### **Interface :**

```
IDestination JMSDestination
{
    get;
    set;
}
```

Extrait et définit la destination du message.

La destination est définie par l'appel `IMessageProducer.send()` lors de l'envoi du message. La valeur de `JMSDestination` est ignorée. Toutefois, vous pouvez utiliser `JMSDestination` pour changer la destination d'un message reçu.

Pour un nouveau message qui n'a pas été envoyé, la méthode renvoie un objet `Destination Null`, sauf si l'application émettrice définit une destination avec `JMSDestination`. Pour un message reçu, la méthode renvoie un objet `Destination` pour la destination qui a été définie par l'appel `IMessageProducer.send()` lorsque le message a été envoyé, sauf si l'application de réception change la destination avec `JMSDestination`.

### **Exceptions :**

- `XMSEException`

## *JMSExpiration - Extraction et définition de JMSExpiration*

### **Interface :**

```
Int64 JMSExpiration
{
    get;
    set;
}
```

Extrait et définit le délai d'expiration du message.

Le délai d'expiration est défini par l'appel `IMessageProducer.send()` lors de l'envoi du message. Sa valeur est calculée en ajoutant la durée de vie, telle que spécifiée par l'application émettrice, à l'heure d'envoi du message. Le délai d'expiration est exprimé en millisecondes, depuis le 1er janvier 1970, 00:00:00 GMT.

Pour un nouveau message qui n'a pas été envoyé, le délai d'expiration est de 0, sauf si l'application émettrice définit un délai d'expiration différent avec `JMSExpiration`. Pour un message reçu, la méthode renvoie le délai d'expiration qui a été défini par l'appel `IMessageProducer.send()` lors de l'envoi du message, sauf si l'application de réception change le délai d'expiration avec `JMSExpiration`.

Si la durée de vie est égale à 0, l'appel `IMessageProducer.send()` définit le délai d'expiration à 0 pour indiquer que le message n'expire pas.

XMS supprime les messages expirés et ne les distribue pas aux applications.

### **Exceptions :**

- `XMSEException`

## *JMSMessageID - Extraction et définition de JMSMessageID*

### **Interface :**

```
String JMSMessageID
{
    get;
    set;
}
```

Extrait et définit l'identificateur du message en tant qu'objet String encapsulant l'identificateur de message.

L'identificateur de message est défini par l'appel `IMessageProducer.send()` lors de l'envoi du message. Pour un message reçu, la méthode renvoie l'identificateur de message qui a été défini par l'appel `IMessageProducer.send()` lors de l'envoi du message, sauf si l'application de réception change l'identificateur de message avec `JMSMessageID`.

Si le message ne contient pas d'identificateur de message, la méthode renvoie une valeur `Null`.

**Exceptions :**

- `XMSEException`

*JMSPriority - Extraction et définition JMSPriority*

**Interface :**

```
Int32 JMSPriority
{
    get;
    set;
}
```

Extrait et définit la priorité du message.

La priorité est définie par l'appel `IMessageProducer.send()` lors de l'envoi du message. La valeur est un entier compris entre 0, priorité la plus faible, et 9, priorité la plus forte.

Pour un nouveau message qui n'a pas été envoyé, la priorité est 4 sauf si l'application émettrice définit une priorité différente avec `JMSPriority`. Pour un message reçu, la méthode renvoie la priorité définie par l'appel `IMessageProducer.send()` lors de l'envoi du message, sauf si l'application de réception change la priorité avec `JMSPriority`.

**Exceptions :**

- `XMSEException`

*JMSRedelivered - Extraction et définition de JMSRedelivered*

**Interface :**

```
Boolean JMSRedelivered
{
    get;
    set;
}
```

Obtient une indication relative à la distribution du message et indique si le message a été redistribué. L'indication est définie par l'appel `IMessageConsumer.receive()` lors de la réception du message.

Cette propriété a les valeurs suivantes :

- `True` si le message est redistribué.
- `False` si le message n'est pas redistribué.

Pour une connexion en temps réel à un courtier, la valeur est toujours `False`.

Une indication de redistribution définie par `JMSRedelivered` avant l'envoi du message est ignorée par l'appel `IMessageProducer.send()` lors de l'envoi du message ; elle est également ignorée et remplacée par l'appel `IMessageConsumer.receive()` lors de la réception du message. Toutefois, vous pouvez utiliser `JMSRedelivered` pour changer l'indication pour un message reçu.

**Exceptions :**

- `XMSEException`

## *JMSReplyTo - Extraction et définition de JMSReplyTo*

### **Interface :**

```
IDestination JMSReplyTo
{
    get;
    set;
}
```

Extrait et définit la destination où une réponse au message doit être envoyée.

La valeur de cette propriété est un objet Destination pour la destination où une réponse doit être envoyée. Un objet Destination Null signifie qu'aucune réponse n'est attendue.

### **Exceptions :**

- XMSEException

## *JMSTimestamp - Extraction et définition de JMSTimestamp*

### **Interface :**

```
Int64 JMSTimestamp
{
    get;
    set;
}
```

Extrait et définit l'heure à laquelle le message a été envoyé.

L'horodatage est défini par l'appel IMessageProducer.send() lorsque le message est envoyé ; il est exprimé en millisecondes, à partir du 1er janvier 1970 00:00:00 GMT.

Pour un nouveau message qui n'a pas été envoyé, l'horodatage est 0, sauf si l'application émettrice définit un horodatage différent avec JMSTimestamp. Pour un message reçu, la méthode renvoie l'horodatage défini par l'appel IMessageProducer.send() lors de l'envoi du message, sauf si l'application de réception change l'horodatage avec JMSTimestamp.

### **Exceptions :**

- XMSEException

### **Remarques :**

1. Si l'horodatage n'est pas défini, la méthode retourne 0, mais n'émet pas d'exception.

## *JMSType - Extraction et définition de JMSType*

### **Interface :**

```
String JMSType
{
    get;
    set;
}
```

Extrait et définit le type du message.

La valeur de JMSType est une chaîne encapsulant le type du message. Si une conversion de données est requise, cette valeur correspond au type après conversion.

### **Exceptions :**

- XMSEException

## *PropertyNames - Extraction des propriétés*

### **Interface :**

```
System.Collections.IEnumerator PropertyNames
{
    get;
}
```

Extrait une énumération des noms de propriétés du message.

### **Exceptions :**

- XMSEException

### **Méthodes**

#### *Acknowledge - Accusé de réception*

### **Interface :**

```
void Acknowledge();
```

Accuse réception de ce message et de tous les messages sans accusé de réception précédents reçus par la session.

Une application peut appeler cette méthode si le mode d'accusé de réception de la session est AcknowledgeMode.ClientAcknowledge. Les appels à la méthode sont ignorés si la session est associée à un autre mode d'accusé de réception ou s'il s'agit d'une session transactionnelle.

Les messages reçus mais pour lesquels aucun accusé de réception n'a été émis peuvent être redistribués.

Pour plus d'information sur l'accusé de réception des messages, voir [../develop/xms\\_cmesack.dita#xms\\_cmesack](#).

### **Paramètres :**

Aucun

### **Retour :**

Vide

### **Exceptions :**

- XMSEException
- IllegalStateException

#### *ClearBody - Suppression du corps*

### **Interface :**

```
void ClearBody();
```

Supprime le corps du message. Les zones d'en-tête et les propriétés du message ne sont pas supprimées.

Si une application supprime le corps d'un message, le corps est conservé sous forme de corps vide dans un nouveau message. L'état d'un corps vide dans un nouveau message dépend du type de corps du message. Pour plus d'informations, voir [Corps d'un message XMS](#).

Une application peut supprimer un corps de message à tout moment, quel que soit son état. Si un corps de message est en lecture seule, l'application doit commencer par le supprimer pour l'écrire ensuite.

### **Paramètres :**

Aucun

**Retour :**

Vide

**Exceptions :**

- XMSEException

*ClearProperties* - Suppression des propriétés

**Interface :**

```
void ClearProperties();
```

Supprime les propriétés du message. Les zones d'en-tête et le corps du message ne sont pas supprimés.

Si une application supprime les propriétés d'un message, les propriétés deviennent lisibles et accessibles en écriture.

Une application peut supprimer les propriétés d'un message à tout moment, quel que soit leur état. Si les propriétés du message sont en lecture seule, l'application doit commencer par les supprimer pour les réécrire ensuite.

**Paramètres :**

Aucun

**Retour :**

Vide

**Exceptions :**

- XMSEException

*PropertyExists* - Vérification de l'existence d'une propriété

**Interface :**

```
Boolean PropertyExists(String propertyName);
```

Vérifie si le message a une propriété avec le nom spécifié.

**Paramètres :****propertyName (entrée)**

Objet String encapsulant le nom de la propriété.

**Retour :**

- `True` si le message a une propriété du nom spécifié.
- `False` si le message n'a pas de propriété avec le nom spécifié.

**Exceptions :**

- XMSEException

**Propriétés et méthodes héritées**

Les méthodes suivantes sont héritées de l'interface [IPropertyContext](#) :

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

## IMessageConsumer

Une application utilise un consommateur de message pour la réception des messages envoyés vers une destination.

### Hiérarchie d'héritage :

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IMessageConsumer
```

Pour obtenir la liste des propriétés XMS définies d'un objet MessageConsumer, voir [«Propriétés de MessageConsumer»](#), à la page 2137.

## .NETpropriétés

*MessageListener - Extraction et définition du programme d'écoute de message*

### Interface :

```
MessageListener MessageListener
{
    get;
    set;
}
```

Extrait le programme d'écoute de message enregistré avec le consommateur de message, puis enregistre un programme d'écoute de message avec le consommateur de message.

Si aucun programme d'écoute de message n'est enregistré avec le consommateur de message, MessageListener est Null. Si un programme d'écoute de message est déjà enregistré avec le consommateur de message, vous pouvez annuler l'enregistrement en spécifiant une valeur null à la place.

Pour plus d'informations sur l'utilisation des programmes d'écoute des messages, voir [Utilisation des programmes d'écoute des messages et des exceptions dans .NET](#).

### Exceptions :

- XMSEException

*MessageSelector - Extraction du sélecteur de message*

### Interface :

```
String MessageSelector
{
    get;
}
```

Extrait le sélecteur de message du consommateur de message. La valeur de retour est un objet de type String encapsulant l'expression de sélecteur de message. Si une conversion de données est requise, cette valeur est l'expression de sélecteur de message après la conversion. Si le consommateur de message n'a pas de sélecteur de message, la valeur de MessageSelector est un objet Null de type String.

### Exceptions :

- XMSEException

## Méthodes

## Close - Fermeture de consommateur de message

### Interface :

```
void Close();
```

Ferme le consommateur de message.

Si une application tente de fermer un consommateur de message qui est déjà fermé, l'appel est ignoré.

### Paramètres :

Aucun

### Retour :

Vide

### Exceptions :

- XMSEException

## Receive - Réception

### Interface :

```
IMessage Receive();
```

Recevez le message suivant pour le consommateur de message. L'appel attend un message sans limite de temps ou jusqu'à ce que le consommateur de message soit fermé.

### Paramètres :

Aucun

### Retour :

Pointeur vers l'objet Message. Si le consommateur de message est fermé pendant que l'appel attend un message, la méthode renvoie un pointeur vers un objet Message de valeur Null.

### Exceptions :

- XMSEException

## Receive - Réception (avec un intervalle d'attente)

### Interface :

```
IMessage Receive(Int64 delay);
```

Recevez le message suivant pour le consommateur de message. L'appel attend un message pendant une période de temps spécifiée ou jusqu'à la fermeture du consommateur de message.

### Paramètres :

#### temps d'attente (entrée)

Temps, en millisecondes, pendant lequel l'appel attend un message. Si vous spécifiez un intervalle d'attente égal à 0, l'appel attend un message sans limite de temps.

### Retour :

Pointeur vers l'objet Message. Si aucun message n'arrive au cours de l'intervalle d'attente, ou si le consommateur de message est fermé alors que l'appel est en attente d'un message, la méthode renvoie un pointeur vers un objet Message de valeur Null, mais n'émet pas d'exception.

### Exceptions :

- XMSEException

*ReceiveNoWait - Réception sans attente*

**Interface :**

```
IMessage ReceiveNoWait();
```

Réception du message suivant pour le consommateur de message, si un message est disponible immédiatement.

**Paramètres :**

Aucun

**Retour :**

Pointeur vers un objet Message. Si aucun message n'est disponible immédiatement, la méthode renvoie un pointeur vers un objet de message Null.

**Exceptions :**

- XMSEException

**Propriétés et méthodes héritées**

Les méthodes suivantes sont héritées de l'interface IPropertyContext :

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

**MessageEOFException**

XMS émet cette exception si XMS rencontre la fin d'un flux de messages d'octets lorsqu'une application lit le corps d'un message d'octets.

**Hiérarchie d'héritage :**

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
|
+----IBM.XMS.MessageEOFException
```

**Propriétés et méthodes héritées**

Les méthodes suivantes sont héritées de l'interface XMSEException :

GetErrorCode, GetLinkedException

**MessageFormatException**

XMS émet cette exception si XMS rencontre un message dont le format n'est pas valide.

**Hiérarchie d'héritage :**

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
|
+----IBM.XMS.MessageFormatException
```

**Propriétés et méthodes héritées**

Les méthodes suivantes sont héritées de l'interface XMSEException :



[GetErrorCode](#), [GetLinkedException](#)

## IMessageListener (délégation)

Une application utilise un programme d'écoute de message pour recevoir des messages de manière asynchrone.

### Hiérarchie d'héritage :

Aucun

### Délégation

*MessageListener* - Programme d'écoute de message

### Interface :

```
public delegate void MessageListener(IMessage msg);
```

Distribue un message à un consommateur de message de manière asynchrone.

Les méthodes qui implémentent cette délégation peuvent être enregistrées avec la connexion.

Pour plus d'informations sur l'utilisation des programmes d'écoute des messages, voir [Utilisation des programmes d'écoute des messages et des exceptions dans .NET](#).

### Paramètres :

#### mesg (entrée)

Objet Message.

### Retour :

Vide

## MessageNotReadableException

XMS émet cette exception si une application tente de lire le corps d'un message en écriture seule.

### Hiérarchie d'héritage :

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
|
+----IBM.XMS.MessageNotReadableException
```

### Propriétés et méthodes héritées

Les méthodes suivantes sont héritées de l'interface [XMSEException](#) :

[GetErrorCode](#), [GetLinkedException](#)

## MessageNotWritableException

XMS émet cette exception si une application tente d'écrire dans le corps d'un message en lecture seule.

### Hiérarchie d'héritage :

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
|
+----IBM.XMS.MessageNotWritableException
```

## Propriétés et méthodes héritées

Les méthodes suivantes sont héritées de l'interface [XMSEException](#) :

[GetErrorCode](#), [GetLinkedException](#)

## IMessageProducer

Une application utilise un expéditeur de message pour envoyer des messages vers une destination.

### Hiérarchie d'héritage :

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IMessageProducer
```

Pour obtenir la liste des propriétés XMS définies d'un objet MessageProducer, voir «[Propriétés de MessageProducer](#)», à la page 2138.

## .NETpropriétés

*DeliveryMode - Extraction et définition du mode de livraison par défaut*

### Interface :

```
DeliveryMode DeliveryMode
{
    get;
    set;
}
```

Extrait et définit le mode de livraison par défaut pour les messages envoyées par l'expéditeur de message.

Le mode de livraison par défaut peut prendre l'une des valeurs suivantes :

`DeliveryMode.Persistent`  
`DeliveryMode.NonPersistent`

Pour une connexion en temps réel à un courtier, la valeur doit être `DeliveryMode.NonPersistent`.

La valeur par défaut est `DeliveryMode.Persistent`, sauf pour une connexion en temps réelle à un courtier, pour laquelle la valeur par défaut est `DeliveryMode.NonPersistent`.

### Exceptions :

- `XMSEException`

*Destination - Définition de la destination*

### Interface :

```
IDestination Destination
{
    get;
}
```

Définit la destination pour l'expéditeur de message.

### Paramètres :

Aucun

### Retour :

Objet `Destination`. Si l'expéditeur de message n'a pas de destination, la méthode renvoie un objet `Destination Null`.

**Exceptions :**

- XMSEException

*DisableMsgID - Extraction et définition de la désactivation de l'identificateur d'ID message*

**Interface :**

```
Boolean DisableMessageID
{
    get;
    set;
}
```

Extrait une indication qui détermine si l'application de réception nécessite que des ID de message soient inclus dans les messages envoyés par l'expéditeur de message.

Sur une connexion à un gestionnaire de files d'attente ou sur une connexion en temps réel à un courtier, cet indicateur est ignoré. Sur une connexion à un bus d'intégration de services, l'indicateur est appliqué.

DisableMsgID peut prendre l'une des valeurs suivantes :

- `True` si l'application de réception ne nécessite pas que des ID message soient inclus dans les messages envoyés par l'expéditeur de message.
- `False` si l'application de réception nécessite que des ID message soient inclus dans les messages envoyés par l'expéditeur de message.

**Exceptions :**

- XMSEException

*DisableMsgTS - Extraction et définition de la désactivation de l'indicateur d'horodatage*

**Interface :**

```
Boolean DisableMessageTimestamp
{
    get;
    set;
}
```

Extrait une indication qui détermine si l'application de réception nécessite que des horodatages soient inclus dans les messages envoyés par l'expéditeur de message.

Sur une connexion en temps réel à un courtier, cet indicateur est ignoré. Sur une connexion à un gestionnaire de files d'attente ou sur une connexion à un bus d'intégration de services, l'indicateur est appliqué.

DisableMsgTS peut prendre l'une des valeurs suivantes :

- `True` si l'application de réception ne nécessite pas que les horodatages soient inclus dans les messages envoyés par l'expéditeur de message.
- `False` si l'application de réception nécessite que les horodatages soient inclus dans les messages envoyés par l'expéditeur de message.

**Retour :****Exceptions :**

- XMSEException

*Priority - Extraction et définition des priorités par défaut*

**Interface :**

```
Int32 Priority
{
```

```
    get;  
    set;  
}
```

Extrait et définit la priorité par défaut pour les messages envoyés par l'expéditeur de message.

La valeur de la priorité de message par défaut est un entier compris entre 0, priorité la plus basse, et 9, priorité la plus élevée.

Sur une connexion en temps réel à un courtier, la priorité d'un message est ignorée.

**Exceptions :**

- XMSEException

*TimeToLive - Extraction et définition de la durée de vie par défaut*

**Interface :**

```
Int64 TimeToLive  
{  
    get;  
    set;  
}
```

Extrait et définit le temps d'existence d'un message avant son expiration.

Le temps est mesuré à partir du moment où l'expéditeur de message envoie le message et représente la durée de vie par défaut (en millisecondes). Une valeur de 0 signifie qu'un message n'expire jamais.

Pour une connexion en temps réel à un courtier, la valeur est toujours 0.

**Exceptions :**

- XMSEException

**Méthodes**

*Close - Fermeture de l'expéditeur de message*

**Interface :**

```
void Close();
```

Ferme l'expéditeur de message.

Si une application tente de fermer un expéditeur de message déjà fermé, l'appel est ignoré.

**Paramètres :**

Aucun

**Retour :**

Vide

**Exceptions :**

- XMSEException

*Send - Envoi*

**Interface :**

```
void Send(IMessage msg) ;
```

Envoyer un message à la destination spécifiée lors de la création de l'expéditeur de message. Le message est envoyé avec le mode de livraison, la priorité et la durée de vie par défaut.

**Paramètres :****msg (entrée)**

Objet Message.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageFormatException
- InvalidDestinationException

*Send - Envoi (avec indication du mode de livraison, de la priorité et de la durée de vie)*

**Interface :**

```
void Send(IMessage msg,
         DeliveryMode deliveryMode,
         Int32 priority,
         Int64 timeToLive);
```

Envoyer un message à la destination spécifiée lors de la création de l'expéditeur de message. Le message est envoyé avec le mode de livraison, la priorité et la durée de vie par défaut.

**Paramètres :****msg (entrée)**

Objet Message.

**deliveryMode (entrée)**

Le mode de livraison du message peut prendre l'une des valeurs suivantes :

DeliveryMode.Persistent  
DeliveryMode.NonPersistent

Pour une connexion en temps réel à un courtier, la valeur doit être DeliveryMode.NonPersistent.

**priority (entrée)**

Priorité du message. La valeur peut être un entier compris entre 0, priorité la plus faible, et 9, priorité la plus forte. Sur une connexion en temps réel à un courtier, la valeur est ignorée.

**timeToLive (entrée)**

Durée de vie du message (en millisecondes). La valeur 0 indique que le message n'arrive jamais à expiration. Pour une connexion en temps réel à un courtier, la valeur doit être 0.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageFormatException
- InvalidDestinationException
- IllegalStateException

*Send - Envoi (vers une destination spécifiée)*

**Interface :**

```
void Send(IDestination dest, IMessage msg) ;
```

Envoyez un message à une destination spécifiée si vous utilisez un expéditeur de message pour lequel aucune destination n'a été spécifiée lors de la création de l'expéditeur de message. Le message est envoyé avec le mode de livraison, la priorité et la durée de vie par défaut.

Généralement, vous indiquez une destination lors de la création de l'expéditeur de message ; si vous ne le faites pas, vous devez spécifier une destination chaque fois que vous envoyez un message.

**Paramètres :**

**dest (entrée)**

Objet Destination.

**msg (entrée)**

Objet Message.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageFormatException
- InvalidDestinationException

*Send - Envoi (vers une destination spécifiée, avec indication du mode de livraison, de la priorité et de la durée de vie)*

**Interface :**

```
void Send(IDestination dest,  
          IMessage msg,  
          DeliveryMode deliveryMode,  
          Int32 priority,  
          Int64 timeToLive) ;
```

Envoyez un message à une destination spécifiée si vous utilisez un expéditeur de message pour lequel aucune destination n'a été spécifiée lors de la création de l'expéditeur de message. Le message est envoyé avec le mode de livraison, la priorité et la durée de vie par défaut.

Généralement, vous indiquez une destination lors de la création de l'expéditeur de message ; si vous ne le faites pas, vous devez spécifier une destination chaque fois que vous envoyez un message.

**Paramètres :**

**dest (entrée)**

Objet Destination.

**msg (entrée)**

Objet Message.

**deliveryMode (entrée)**

Le mode de livraison du message peut prendre l'une des valeurs suivantes :

DeliveryMode.Persistent  
DeliveryMode.NonPersistent

Pour une connexion en temps réel à un courtier, la valeur doit être DeliveryMode.NonPersistent.

**priority (entrée)**

Priorité du message. La valeur peut être un entier compris entre 0, priorité la plus faible, et 9, priorité la plus forte. Sur une connexion en temps réel à un courtier, la valeur est ignorée.

**timeToLive (entrée)**

Durée de vie du message (en millisecondes). La valeur 0 indique que le message n'arrive jamais à expiration. Pour une connexion en temps réel à un courtier, la valeur doit être 0.

**Retour :**

Vide

**Exceptions :**

- XMSException
- MessageFormatException
- InvalidDestinationException
- IllegalStateException

**Propriétés et méthodes héritées**

Les méthodes suivantes sont héritées de l'interface IPropertyContext :

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

**IOBJECTMESSAGE**

Un message d'objet est un message dont le corps comprend un objet Java ou .NET sérialisé.

**Hiérarchie d'héritage :**

```

IBM.XMS.IPropertyContext
|
+---- IBM.XMS.IMessage
      |
      +---- IBM.XMS.IObjectMessage
  
```

**.NETpropriétés**

*Objet - Extraction et définition de l'objet sous forme d'octets*

**Interface :**

```

System.Object Object
{
    get;
    set;
}

Byte[] GetObject();
  
```

Extrait et définit l'objet qui forme le corps du message d'objet.

**Exceptions :**

- XMSException
- MessageNotReadableException
- MessageEOFException
- MessageNotWritableException

**Propriétés et méthodes héritées**

Les propriétés suivantes sont héritées de l'interface IMessage :

JMSCorrelationID, JMSDeliveryMode, JMSDestination, JMSExpiration, JMSMessageID, JMSPriority, JMSRedistribuée, JMSReplyTo, JMSTimestamp, JMSType, Propriétés

Les méthodes suivantes sont héritées de l'interface IMessage :

[clearBody](#), [clearProperties](#), [PropertyExists](#)

Les méthodes suivantes sont héritées de l'interface `IPropertyContext` :

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

## **IPropertyContext**

`IPropertyContext` est une superclasse abstraite qui contient des méthodes permettant d'obtenir et de définir des propriétés. Ces méthodes sont héritées par d'autres classes.

### **Hiérarchie d'héritage :**

Aucun

### **Méthodes**

*GetBooleanProperty* – Extraction de la propriété booléenne

#### **Interface :**

```
Boolean GetBooleanProperty(String property_name);
```

extrait la valeur de la propriété booléenne avec le nom spécifié.

#### **Paramètres :**

##### **property\_name (entrée)**

Objet `String` encapsulant le nom de la propriété.

#### **Retour :**

Valeur de la propriété.

#### **Contexte d'unité d'exécution :**

Déterminé par la sous-classe

#### **Exceptions :**

- `XMSEException`

*GetByteProperty* – Extraction de la propriété d'octet

#### **Interface :**

```
Byte    GetByteProperty(String property_name) ;  
Int16   GetSignedByteProperty(String property_name) ;
```

Extrait la valeur de la propriété d'octet identifiée par nom.

#### **Paramètres :**

##### **property\_name (entrée)**

Objet `String` encapsulant le nom de la propriété.

#### **Retour :**

Valeur de la propriété.

#### **Contexte d'unité d'exécution :**

Déterminé par la sous-classe

#### **Exceptions :**

- `XMSEException`



*GetBytesProperty – Extraction de la propriété de tableau d'octets*

**Interface :**

```
Byte[] GetBytesProperty(String property_name) ;
```

Extrait la valeur du tableau d'octets identifié par nom.

**Paramètres :**

**property\_name (entrée)**  
Objet String encapsulant le nom de la propriété.

**Retour :**  
Nombre d'octets dans le tableau.

**Contexte d'unité d'exécution :**  
Déterminé par la sous-classe

**Exceptions :**  
• XMSEException

*GetCharProperty – Extraction de la propriété de caractère*

**Interface :**

```
Char GetCharProperty(String property_name) ;
```

Extrait la valeur de la propriété de caractère sur 2 octets identifiée par nom.

**Paramètres :**

**property\_name (entrée)**  
Objet String encapsulant le nom de la propriété.

**Retour :**  
Valeur de la propriété.

**Contexte d'unité d'exécution :**  
Déterminé par la sous-classe

**Exceptions :**  
• XMSEException

*GetDoubleProperty – Extraction de la propriété de nombre en virgule flottante à double précision*

**Interface :**

```
Double GetDoubleProperty(String property_name) ;
```

Extrait la valeur de la propriété du nombre en virgule flottante à double précision identifiée par nom.

**Paramètres :**

**property\_name (entrée)**  
Objet String encapsulant le nom de la propriété.

**Retour :**  
Valeur de la propriété.

**Contexte d'unité d'exécution :**  
Déterminé par la sous-classe

**Exceptions :**  
• XMSEException

### *GetFloatProperty – Extraction de la propriété de virgule flottante*

#### **Interface :**

```
Single GetFloatProperty(String property_name) ;
```

Extrait la valeur de la propriété de virgule flottante identifiée par nom.

#### **Paramètres :**

##### **property\_name (entrée)**

Objet String encapsulant le nom de la propriété.

#### **Retour :**

Valeur de la propriété.

#### **Contexte d'unité d'exécution :**

Déterminé par la sous-classe

#### **Exceptions :**

- XMSEException

### *GetIntProperty – Extraction de la propriété d'entier*

#### **Interface :**

```
Int32 GetIntProperty(String property_name) ;
```

Extrait la valeur de la propriété d'entier identifiée par nom.

#### **Paramètres :**

##### **property\_name (entrée)**

Objet String encapsulant le nom de la propriété.

#### **Retour :**

Valeur de la propriété.

#### **Contexte d'unité d'exécution :**

Déterminé par la sous-classe

#### **Exceptions :**

- XMSEException

### *GetLongProperty – Extraction de la propriété d'entier long*

#### **Interface :**

```
Int64 GetLongProperty(String property_name) ;
```

Extrait la valeur de la propriété d'entier long identifiée par nom.

#### **Paramètres :**

##### **property\_name (entrée)**

Objet String encapsulant le nom de la propriété.

#### **Retour :**

Valeur de la propriété.

#### **Contexte d'unité d'exécution :**

Déterminé par la sous-classe

#### **Exceptions :**

- XMSEException

*GetObjectProperty – Extraction de la propriété d'objet*

**Interface :**

```
Object GetObjectProperty( String property_name) ;
```

Extrait la valeur et le type de données de la propriété identifiée par nom.

**Paramètres :**

**property\_name (entrée)**

Objet String encapsulant le nom de la propriété.

**Retour :**

Valeur de la propriété, dont le type d'objet peut être l'un des suivants :

Boolean  
Byte  
Byte[]  
Char  
Double  
Single  
Int32  
Int64  
Int16  
String

**Contexte d'unité d'exécution :**

Déterminé par la sous-classe

**Exceptions :**

- XMSEException

*GetShortProperty – Extraction de la propriété d'entier court*

**Interface :**

```
Int16 GetShortProperty(String property_name) ;
```

Extrait la valeur de la propriété d'entier court identifiée par nom.

**Paramètres :**

**property\_name (entrée)**

Objet String encapsulant le nom de la propriété.

**Retour :**

Valeur de la propriété.

**Contexte d'unité d'exécution :**

Déterminé par la sous-classe

**Exceptions :**

- XMSEException

*GetStringProperty – Extraction de la propriété de chaîne*

**Interface :**

```
String GetStringProperty(String property_name) ;
```

Extrait la valeur de la chaîne identifiée par nom.

**Paramètres :****property\_name (entrée)**

Objet String encapsulant le nom de la propriété.

**Retour :**

Objet String encapsulant la chaîne, c'est-à-dire la valeur de la propriété. Si une conversion de données est requise, cette valeur correspond à la chaîne après conversion.

**Contexte d'unité d'exécution :**

Déterminé par la sous-classe

**Exceptions :**

- XMSEException

*SetBooleanProperty – Définition de la propriété booléenne*

**Interface :**

```
void SetBooleanProperty( String property_name, Boolean value) ;
```

Définit la valeur de la propriété booléenne identifiée par nom.

**Paramètres :****property\_name (entrée)**

Objet String encapsulant le nom de la propriété.

**value (entrée)**

Valeur de la propriété.

**Retour :**

Vide

**Contexte d'unité d'exécution :**

Déterminé par la sous-classe

**Exceptions :**

- XMSEException
- MessageNotWritableException

*SetByteProperty – Définition de la propriété d'octet*

**Interface :**

```
void SetByteProperty( String property_name, Byte value) ;  
void SetSignedByteProperty( String property_name, Int16 value) ;
```

Définit la valeur de la propriété d'octet identifiée par nom.

**Paramètres :****property\_name (entrée)**

Objet String encapsulant le nom de la propriété.

**value (entrée)**

Valeur de la propriété.

**Retour :**

Vide

**Contexte d'unité d'exécution :**

Déterminé par la sous-classe

**Exceptions :**

- XMSEException

- MessageNotWritableException

*SetBytesProperty – Définition de la propriété de tableau d'octets*

**Interface :**

```
void SetBytesProperty( String property_name, Byte[] value ) ;
```

Définit la valeur de la propriété de tableau d'octets identifiée par nom.

**Paramètres :**

**property\_name (entrée)**

Objet String encapsulant le nom de la propriété.

**value (entrée)**

Valeur de la propriété qui correspond à un tableau d'octets.

**Retour :**

Vide

**Contexte d'unité d'exécution :**

Déterminé par la sous-classe

**Exceptions :**

- XMSEException
- MessageNotWritableException

*SetCharProperty – Définition de la propriété de caractère*

**Interface :**

```
void SetCharProperty( String property_name, Char value) ;
```

Définit la valeur de la propriété de caractère sur 2 octets identifiée par nom.

**Paramètres :**

**property\_name (entrée)**

Objet String encapsulant le nom de la propriété.

**value (entrée)**

Valeur de la propriété.

**Retour :**

Vide

**Contexte d'unité d'exécution :**

Déterminé par la sous-classe

**Exceptions :**

- XMSEException
- MessageNotWritableException

*SetDoubleProperty – Définition de la propriété de nombre en virgule flottante à double précision*

**Interface :**

```
void SetDoubleProperty( String property_name, Double value) ;
```

Définit la valeur de la propriété du nombre en virgule flottante à double précision identifiée par nom.

**Paramètres :****property\_name (entrée)**

Objet String encapsulant le nom de la propriété.

**value (entrée)**

Valeur de la propriété.

**Retour :**

Vide

**Contexte d'unité d'exécution :**

Déterminé par la sous-classe

**Exceptions :**

- XMSEException
- MessageNotWritableException

*SetFloatProperty – Définition de la propriété de virgule flottante*

**Interface :**

```
void SetFloatProperty( String property_name, Single value) ;
```

Définit la valeur de la propriété de virgule flottante identifiée par nom.

**Paramètres :****property\_name (entrée)**

Objet String encapsulant le nom de la propriété.

**value (entrée)**

Valeur de la propriété.

**Retour :**

Vide

**Contexte d'unité d'exécution :**

Déterminé par la sous-classe

**Exceptions :**

- XMSEException
- MessageNotWritableException

*SetIntProperty – Définition de la propriété d'entier*

**Interface :**

```
void SetIntProperty( String property_name, Int32 value) ;
```

Définit la valeur de la propriété d'entier identifiée par nom.

**Paramètres :****property\_name (entrée)**

Objet String encapsulant le nom de la propriété.

**value (entrée)**

Valeur de la propriété.

**Retour :**

Vide

**Contexte d'unité d'exécution :**

Déterminé par la sous-classe

**Exceptions :**

- XMSEException
- MessageNotWritableException

*SetLongProperty – Définition de la propriété d'entier long*

**Interface :**

```
void SetLongProperty( String property_name, Int64 value) ;
```

Définit la valeur de la propriété d'entier long identifiée par nom.

**Paramètres :****property\_name (entrée)**

Objet String encapsulant le nom de la propriété.

**value (entrée)**

Valeur de la propriété.

**Retour :**

Vide

**Contexte d'unité d'exécution :**

Déterminé par la sous-classe

**Exceptions :**

- XMSEException
- MessageNotWritableException

*SetObjectProperty – Définition de la propriété d'objet*

**Interface :**

```
void SetObjectProperty( String property_name, Object value) ;
```

Définit la valeur et le type de données de la propriété identifiée par nom.

**Paramètres :****property\_name (entrée)**

Objet String encapsulant le nom de la propriété.

**objectType (entrée)**

Valeur de la propriété, dont le type d'objet peut être l'un des suivants :

Boolean  
Byte  
Byte[]  
Char  
Double  
Single  
Int32  
Int64  
Int16  
String

**value (entrée)**

Valeur de la propriété qui correspond à un tableau d'octets.

**length (entrée)**

Nombre d'octets dans le tableau.

**Retour :**

Vide

**Contexte d'unité d'exécution :**

Déterminé par la sous-classe

**Exceptions :**

- XMSEException
- MessageNotWritableException

*SetShortProperty – Définition de la propriété d'entier court*

**Interface :**

```
void SetShortProperty( String property_name, Int16 value) ;
```

Définit la valeur de la propriété d'entier court identifiée par nom.

**Paramètres :****property\_name (entrée)**

Objet String encapsulant le nom de la propriété.

**value (entrée)**

Valeur de la propriété.

**Retour :**

Vide

**Contexte d'unité d'exécution :**

Déterminé par la sous-classe

**Exceptions :**

- XMSEException
- MessageNotWritableException

*SetStringProperty – Définition de la propriété de chaîne*

**Interface :**

```
void SetStringProperty( String property_name, String value);
```

Définit la valeur de la propriété de chaîne identifiée par nom.

**Paramètres :****property\_name (entrée)**

Objet String encapsulant le nom de la propriété.

**value (entrée)**

Objet String encapsulant la chaîne, c'est-à-dire la valeur de la propriété.

**Retour :**

Vide

**Contexte d'unité d'exécution :**

Déterminé par la sous-classe

**Exceptions :**

- XMSEException
- MessageNotWritableException



## IQueueBrowser

Une application utilise un navigateur de files d'attente pour parcourir les messages sur une file d'attente sans les supprimer.

### Hiérarchie d'héritage :

```
IBM.XMS.IPropertyContext
System.Collections.IEnumerable
|
+---- IBM.XMS.IQueueBrowser
```

## .NETpropriétés

*MessageSelector - Extraction du sélecteur de message*

### Interface :

```
String MessageSelector
{
    get;
}
```

Extrait le sélecteur de message pour le navigateur de files d'attente.

Le sélecteur de message est un objet de type String encapsulant l'expression de sélecteur de message. Si une conversion de données est requise, cette valeur est l'expression de sélecteur de message après la conversion. Si le navigateur de files d'attente n'a pas de sélecteur de message, la méthode renvoie un objet de type String Null.

### Exceptions :

- XMSEException

*Queue - Extraction de file d'attente*

### Interface :

```
IDestination Queue
{
    get;
}
```

extrait la file d'attente associée au navigateur de files d'attente en tant qu'objet de destination représentant la file d'attente.

### Exceptions :

- XMSEException

## Méthodes

*Close - Fermeture du navigateur de files d'attente*

### Interface :

```
void Close();
```

Ferme le navigateur de files d'attente.

Si une application tente de fermer un navigateur de files d'attente qui est déjà fermé, l'appel est ignoré.

### Paramètres :

Aucun

**Retour :**

Vide

**Exceptions :**

- XMSEException

*GetEnumerator - Extraction de messages*

**Interface :**

```
IEnumerator GetEnumerator();
```

Extrait une liste des messages de la file d'attente.

La méthode renvoie une expression énumérative qui encapsule une liste d'objets Message. L'ordre des objets Message est le même que celui dans lequel les messages sont extraits de la file d'attente.

L'application peut alors utiliser l'expression énumérative pour parcourir chaque message à son tour.

L'expression énumérative est mise à jour de manière dynamique à mesure que les messages sont insérés dans la file d'attente et supprimés de la file d'attente. Chaque fois que l'application appelle `IEnumerator.MoveNext()` pour parcourir le message suivant dans la file d'attente, le message reflète le contenu en cours de la file d'attente.

Si une application appelle cette méthode plus d'une fois pour un navigateur de files d'attente, chaque appel renvoie une nouvelle expression énumérative. L'application peut ainsi utiliser plus d'une expression énumérative pour parcourir les messages dans une file d'attente et gérer plusieurs positions dans la file d'attente.

**Paramètres :**

Aucun

**Retour :**

Objet `Iterator`.

**Exceptions :**

- XMSEException

**Propriétés et méthodes héritées**

Les méthodes suivantes sont héritées de l'interface [IPropertyContext](#) :

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

**Demandeur**

Une application utilise un demandeur pour envoyer un message de demande, puis attendre et recevoir la réponse.

**Hiérarchie d'héritage :**

Aucun

**Constructeurs**

*Requestor - Création d'un demandeur*

**Interface :**

```
Requestor(ISession sess, IDestination dest);
```

Crée un demandeur.

**Paramètres :**

**sess (entrée)**

Objet Session. La session ne doit pas être transactionnelle et doit être associée à l'un des modes d'accusé de réception suivants :

AcknowledgeMode.AutoAcknowledge  
AcknowledgeMode.DupsOkAcknowledge

**dest (entrée)**

Objet Destination représentant la destination vers laquelle l'application peut envoyer des messages de demande.

**Contexte d'unité d'exécution :**

Session associée au demandeur

**Exceptions :**

- XMSException

**Méthodes**

*Close – Fermeture du demandeur*

**Interface :**

```
void Close();
```

Ferme le demandeur.

Si une application tente de fermer un demandeur déjà fermé, l'appel est ignoré.

**Remarque :** Lorsqu'une application ferme un demandeur, la session associée n'est pas fermée. A cet égard, XMS se comporte différemment de JMS.

**Paramètres :**

Aucun

**Retour :**

Vide

**Contexte d'unité d'exécution :**

Tous

**Exceptions :**

- XMSException

*Request – Réponse à une demande*

**Interface :**

```
IMessage Request(IMessage requestMessage);
```

Envoie un message de demande, puis attend et reçoit une réponse de l'application ayant reçu le message de demande.

Un appel à cette demande bloque jusqu'à la réception d'une réponse ou la fin de session.

**Paramètres :**

**requestMessage (entrée)**

Objet Message encapsulant le message de demande.

**Retour :**

Pointeur vers l'objet Message encapsulant le message de demande.

**Contexte d'unité d'exécution :**

Session associée au demandeur

**Exceptions :**

- XMSEException

**ResourceAllocationException**

XMS émet cette exception si XMS ne peut pas allouer les ressources requises par une méthode.

**Hiérarchie d'héritage :**

```

IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
      |
      +----IBM.XMS.ResourceAllocationException
  
```

**Propriétés et méthodes héritées**

Les méthodes suivantes sont héritées de l'interface [XMSEException](#) :

[GetErrorCode](#), [GetLinkedException](#)

**SecurityException**

XMS émet cette exception si l'ID utilisateur et le mot de passe fournis pour authentifier une application sont rejetés. XMS émet également cette exception si une vérification des droits échoue et empêche l'aboutissement d'une méthode.

**Hiérarchie d'héritage :**

```

IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
      |
      +----IBM.XMS.SecurityException
  
```

**Propriétés et méthodes héritées**

Les méthodes suivantes sont héritées de l'interface [XMSEException](#) :

[GetErrorCode](#), [GetLinkedException](#)

**ISession**

Une session est un contexte à unité d'exécution unique pour l'envoi et la réception de messages.

**Hiérarchie d'héritage :**

```

IBM.XMS.IPropertyContext
|
+----IBM.XMS.ISession
  
```

Pour obtenir la liste des propriétés XMS définies d'un objet Session, voir [«Propriétés de Session»](#), à la page 2138.

**.NETpropriétés**

*AcknowledgeMode - Extraction du mode d'accusé de réception*

## Interface :

```
AcknowledgeMode AcknowledgeMode
{
    get;
}
```

Extrait le mode d'accusé de réception pour la session.

Le mode d'accusé de réception est spécifié lors de la création de la session.

Si la session n'est pas transactionnelle, le mode d'accusé de réception correspond à l'une des valeurs suivantes :

```
AcknowledgeMode.AutoAcknowledge
AcknowledgeMode.ClientAcknowledge
AcknowledgeMode.DupsOkAcknowledge
```

Pour plus d'informations sur les modes d'accusé de réception, voir [Accusé de réception de message](#).

Une session transactionnelle n'a pas de mode d'accusé de réception. Si la session est transactionnelle, la méthode renvoie la valeur `AcknowledgeMode.SessionTransacted`.

## Exceptions :

- `XMSEException`

*Transacted* - Détermine s'il s'agit d'une session transactionnelle

## Interface :

```
Boolean Transacted
{
    get;
}
```

Détermine si la session est transactionnelle.

L'état transactionnel peut être :

- True, si la session est transactionnelle.
- False, si la session n'est pas transactionnelle.

Pour une connexion en temps réel à un courtier, la méthode renvoie toujours False.

## Exceptions :

- `XMSEException`

## Méthodes

*Close* - Fermeture de session

## Interface :

```
void Close();
```

Ferme la session. Si la session est transactionnelle, toute transaction en cours est annulée.

Si une application tente de fermer une session déjà fermée, l'appel est ignoré.

## Paramètres :

Aucun

## Retour :

Vide

**Contexte d'unité d'exécution :**

Tous

**Exceptions :**

- XMSEException

*Commit - Validation*

**Interface :**

```
void Commit();
```

Valide tous les messages traités lors de la transaction en cours.

La session doit être transactionnelle.

**Paramètres :**

Aucun

**Retour :**

Vide

**Exceptions :**

- XMSEException
- IllegalStateException
- TransactionRolledBackException

*CreateBrowser - Création d'un navigateur de files d'attente*

**Interface :**

```
IQueueBrowser CreateBrowser(IDestination queue) ;
```

Crée un navigateur de files d'attente pour la file d'attente spécifiée.

**Paramètres :****queue (entrée)**

Objet Destination représentant la file d'attente.

**Retour :**

Objet QueueBrowser.

**Exceptions :**

- XMSEException
- InvalidDestinationException

*CreateBrowser - Création d'un navigateur de files d'attente (avec sélecteur de message)*

**Interface :**

```
IQueueBrowser CreateBrowser(IDestination queue, String selector) ;
```

Création d'un navigateur de files d'attente pour la file d'attente spécifiée avec un sélecteur de message.

**Paramètres :****queue (entrée)**

Objet Destination représentant la file d'attente.

**selector (entrée)**

Objet String encapsulant une expression de sélecteur de message. Seuls les messages dont les propriétés correspondent à l'expression du sélecteur de message sont distribués sur le navigateur de files d'attente.

Un objet String Null signifie qu'aucun sélecteur de message n'est défini pour le navigateur de files d'attente.

**Retour :**

Objet QueueBrowser.

**Exceptions :**

- XMSEException
- InvalidDestinationException
- InvalidSelectorException

*CreateBytesMessage - Création d'un message d'octets*

**Interface :**

```
IBytesMessage CreateBytesMessage();
```

Crée un message d'octets.

**Paramètres :**

Aucun

**Retour :**

Objet BytesMessage.

**Exceptions :**

- XMSEException
- IllegalStateException (The session is closed)

*CreateConsumer - Création d'un consommateur*

**Interface :**

```
IMessageConsumer CreateConsumer(IDestination dest) ;
```

Crée un consommateur de message pour la destination spécifiée.

**Paramètres :****dest (entrée)**

Objet Destination.

**Retour :**

Objet MessageConsumer.

**Exceptions :**

- XMSEException
- InvalidDestinationException

*CreateConsumer - Création d'un consommateur (avec sélecteur de message)*

**Interface :**

```
IMessageConsumer CreateConsumer(IDestination dest,  
                                String selector) ;
```

Crée un consommateur de message pour la destination spécifiée avec un sélecteur de message.

**Paramètres :****dest (entrée)**

Objet Destination.

**selector (entrée)**

Objet String encapsulant une expression de sélecteur de message. Seuls les messages dont les propriétés correspondent à l'expression du sélecteur de message sont distribués sur le consommateur de message.

Un objet String Null signifie qu'aucun sélecteur de message n'est défini pour le consommateur de message.

**Retour :**

Objet MessageConsumer.

**Exceptions :**

- XMSEException
- InvalidDestinationException
- InvalidSelectorException

*CreateConsumer - Création d'un consommateur (avec sélecteur de message et indicateur de message local)*

**Interface :**

```
IMessageConsumer CreateConsumer(IDestination dest,  
                                String selector,  
                                Boolean noLocal) ;
```

Crée un consommateur de message pour la destination spécifiée avec un sélecteur de message et, si la destination est une rubrique, spécifie si le consommateur de message reçoit les messages publiés par sa propre connexion.

**Paramètres :****dest (entrée)**

Objet Destination.

**selector (entrée)**

Objet String encapsulant une expression de sélecteur de message. Seuls les messages dont les propriétés correspondent à l'expression du sélecteur de message sont distribués sur le consommateur de message.

Un objet String Null signifie qu'aucun sélecteur de message n'est défini pour le consommateur de message.

**noLocal (entrée)**

La valeur True signifie que le consommateur de message ne reçoit pas les messages publiés par sa propre connexion. La valeur False signifie que le consommateur de message reçoit les messages publiés par sa propre connexion. La valeur par défaut est False.

**Retour :**

Objet MessageConsumer.

**Exceptions :**

- XMSEException
- InvalidDestinationException
- InvalidSelectorException



## *CreateDurableSubscriber - Création d'un abonné durable*

### **Interface :**

```
IMessageConsumer CreateDurableSubscriber(IDestination dest,  
                                         String subscription) ;
```

Crée un abonné durable pour la rubrique spécifiée.

Cette méthode n'est pas valide pour une connexion en temps réel à un courtier.

Pour plus d'informations sur les abonnés durables, voir [Abonnés durables](#).

### **Paramètres :**

#### **dest (entrée)**

Un objet Destination représentant la rubrique. Il ne doit pas s'agir d'une rubrique temporaire.

#### **subscription (entrée)**

Un objet String encapsulant un nom qui identifie l'abonnement durable. Le nom doit être unique dans l'identificateur de client pour la connexion.

### **Retour :**

Objet MessageConsumer représentant l'abonné durable.

### **Exceptions :**

- XMSEException
- InvalidDestinationException

## *CreateDurableSubscriber - Création d'un abonné durable (avec sélecteur de message et indicateur de message local)*

### **Interface :**

```
IMessageConsumer CreateDurableSubscriber(IDestination dest,  
                                         String subscription,  
                                         String selector,  
                                         Boolean noLocal) ;
```

Crée un abonné durable pour la rubrique spécifiée avec un sélecteur de message et en indiquant si l'abonné durable reçoit les messages publiés par sa propre connexion.

Cette méthode n'est pas valide pour une connexion en temps réel à un courtier.

Pour plus d'informations sur les abonnés durables, voir [Abonnés durables](#).

### **Paramètres :**

#### **dest (entrée)**

Un objet Destination représentant la rubrique. Il ne doit pas s'agir d'une rubrique temporaire.

#### **subscription (entrée)**

Un objet String encapsulant un nom qui identifie l'abonnement durable. Le nom doit être unique dans l'identificateur de client pour la connexion.

#### **selector (entrée)**

Objet String encapsulant une expression de sélecteur de message. Seuls les messages dont les propriétés correspondent à l'expression du sélecteur de message sont distribués sur l'abonné durable.

Un objet String Null signifie qu'aucun sélecteur de message n'est défini pour l'abonné durable.

#### **noLocal (entrée)**

La valeur True signifie que l'abonné durable ne reçoit pas les messages publiés par sa propre connexion. La valeur False signifie que l'abonné durable reçoit les messages publiés par sa propre connexion. La valeur par défaut est False.

**Retour :**

Objet MessageConsumer représentant l'abonné durable.

**Exceptions :**

- XMSEException
- InvalidDestinationException
- InvalidSelectorException

*CreateMapMessage - Création d'un message de mappe*

**Interface :**

```
IMapMessage CreateMapMessage();
```

Crée un message de mappe.

**Paramètres :**

Aucun

**Retour :**

Objet MapMessage.

**Exceptions :**

- XMSEException
- IllegalStateException (The session is closed)

*CreateMessage - Création d'un message*

**Interface :**

```
IMessage CreateMessage();
```

Crée un message sans corps.

**Paramètres :**

Aucun

**Retour :**

Objet Message.

**Exceptions :**

- XMSEException
- IllegalStateException (The session is closed)

*CreateObjectMessage - Création d'un message d'objet*

**Interface :**

```
IObjectMessage CreateObjectMessage();
```

Crée un message d'objet.

**Paramètres :**

Aucun

**Retour :**

Objet ObjectMessage.

**Exceptions :**

- XMSEException
- IllegalStateException (The session is closed)

### *CreateProducer - Création d'un expéditeur*

#### **Interface :**

```
IMessageProducer CreateProducer(IDestination dest) ;
```

Crée un expéditeur de message pour envoyer des messages vers la destination spécifiée.

#### **Paramètres :**

##### **dest (entrée)**

Objet Destination.

Si vous spécifiez un objet Destination Null, l'expéditeur de message est créé sans destination.

Dans ce cas, l'application doit spécifier une destination chaque fois qu'elle utilise l'expéditeur de message pour envoyer un message.

#### **Retour :**

Objet MessageProducer.

#### **Exceptions :**

- XMSEException
- InvalidDestinationException

### *CreateQueue - Création d'une file d'attente*

#### **Interface :**

```
IDestination CreateQueue(String queue) ;
```

Crée un objet Destination pour représenter la file d'attente sur le serveur de messagerie.

Cette méthode ne permet pas de créer la file d'attente sur le serveur de messagerie. Vous devez créer celle-ci avant qu'une application puisse appeler cette méthode.

#### **Paramètres :**

##### **queue (entrée)**

Un objet de type String encapsulant le nom de la file d'attente ou un identificateur URI (Uniform Resource Identifier) qui identifie la file d'attente.

#### **Retour :**

Objet Destination représentant la file d'attente.

#### **Exceptions :**

- XMSEException

### *CreateStreamMessage - Création d'un message de flux*

#### **Interface :**

```
IStreamMessage CreateStreamMessage();
```

Crée un message de flux.

#### **Paramètres :**

Aucun

#### **Retour :**

Objet StreamMessage.

#### **Exceptions :**

- XMSEException
- XMS\_ILLEGAL\_STATE\_EXCEPTION

### *CreateTemporaryQueue - Création d'une file d'attente temporaire*

#### **Interface :**

```
IDestination CreateTemporaryQueue() ;
```

Crée une file d'attente temporaire.

La portée de la file d'attente temporaire est la connexion. Seules les sessions créées par la connexion peuvent utiliser la file d'attente temporaire.

La file d'attente temporaire reste active jusqu'à sa suppression explicite ou la fin de la connexion.

Pour plus d'informations sur les files d'attente temporaires, voir [Destinations temporaires](#).

#### **Paramètres :**

Aucun

#### **Retour :**

Objet Destination représentant la file d'attente temporaire.

#### **Exceptions :**

- XMSEException

### *CreateTemporaryTopic - Création d'une rubrique temporaire*

#### **Interface :**

```
IDestination CreateTemporaryTopic() ;
```

Crée une rubrique temporaire.

La portée de la rubrique temporaire est la connexion. Seules les sessions créées par la connexion peuvent utiliser la rubrique temporaire.

La rubrique temporaire reste active jusqu'à sa suppression explicite ou la fin de la connexion.

Pour plus d'informations sur les rubriques temporaires, voir [Destinations temporaires](#).

#### **Paramètres :**

Aucun

#### **Retour :**

Objet Destination représentant la rubrique temporaire.

#### **Exceptions :**

- XMSEException

### *CreateTextMessage - Création d'un message texte*

#### **Interface :**

```
ITextMessage CreateTextMessage() ;
```

Crée un message texte avec un corps vide.

#### **Paramètres :**

Aucun

#### **Retour :**

Objet TextMessage.

#### **Exceptions :**

- XMSEException

*CreateTextMessage - Création d'un message texte (initialisé)*

**Interface :**

```
ITextMessage CreateTextMessage(String initialValue);
```

Crée un message texte dont le corps est initialisé avec le texte spécifié.

**Paramètres :**

**initialValue (entrée)**

Objet String encapsulant le texte qui initialise le corps du message texte.

Aucun

**Retour :**

Objet TextMessage.

**Exceptions :**

- XMSEException

*CreateTopic - Création d'une rubrique*

**Interface :**

```
IDestination CreateTopic(String topic) ;
```

Crée un objet Destination pour représenter une rubrique.

**Paramètres :**

**topic (entrée)**

Objet String encapsulant le nom de la rubrique ou un identificateur URI qui identifie la rubrique.

**Retour :**

Objet Destination représentant la rubrique.

**Exceptions :**

- XMSEException

*Recover - Restauration*

**Interface :**

```
void Recover();
```

Restaure la session. La livraison du message est arrêtée puis reprise avec le message le plus ancien sans accusé de réception.

La session ne doit pas être transactionnelle.

Pour plus d'informations sur la récupération d'une session, voir [Accusé de réception de message](#).

**Paramètres :**

Aucun

**Retour :**

Vide

**Exceptions :**

- XMSEException
- IllegalStateException

## *Rollback - Annulation*

### **Interface :**

```
void Rollback();
```

Annule tous les messages traités lors de la transaction en cours.

La session doit être transactionnelle.

### **Paramètres :**

Aucun

### **Retour :**

Vide

### **Exceptions :**

- XMSEException
- IllegalStateException

## *Unsubscribe - Désabonnement*

### **Interface :**

```
void Unsubscribe(String subscription);
```

Supprime un abonnement durable. Le serveur de messagerie supprime l'enregistrement de l'abonnement durable qu'il gère et n'envoie plus de messages à l'abonné durable.

Une application ne peut pas supprimer un abonnement durable dans les circonstances suivantes :

- Un consommateur de message est actif pour l'abonnement durable
- Un message consommé fait partie d'une transaction en attente
- Aucun accusé de réception n'a été émis pour un message consommé

Cette méthode n'est pas valide pour une connexion en temps réel à un courtier.

### **Paramètres :**

#### **subscription (entrée)**

Objet String encapsulant le nom qui identifie l'abonnement durable.

### **Retour :**

Vide

### **Exceptions :**

- XMSEException
- InvalidDestinationException
- IllegalStateException

## ***Propriétés et méthodes héritées***

Les méthodes suivantes sont héritées de l'interface [IPropertyContext](#) :

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

## IStreamMessage

Un message de flux est un message dont le corps contient un flux de valeurs et où chaque valeur a un type de données associé. Le contenu du corps est écrit et lu séquentiellement.

### Hiérarchie d'héritage :

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IMessage
|
+----IBM.XMS.IStreamMessage
```

Lorsqu'une application lit une valeur à partir du flux de message, celle-ci est convertie par XMS dans un autre type de données. Pour plus d'informations sur cette forme de conversion implicite, voir [Corps d'un message XMS](#).

### Méthodes

*ReadBoolean* - Lecture de valeur booléenne

#### Interface :

```
Boolean ReadBoolean();
```

Lit une valeur booléenne depuis le flux de message.

#### Paramètres :

Aucun

#### Retour :

Valeur booléenne lue.

#### Exceptions :

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadByte* - Lecture d'un octet

#### Interface :

```
Int16  ReadSignedByte();
Byte   ReadByte();
```

Lit un entier signé de 8 bits depuis le flux de message.

#### Paramètres :

Aucun

#### Retour :

Octet lu.

#### Exceptions :

- XMSException
- MessageNotReadableException
- MessageEOFException

## *ReadBytes - Lecture d'octets*

### **Interface :**

```
Int32 ReadBytes(Byte[] array);
```

Lit un tableau d'octets depuis le flux de message.

### **Paramètres :**

#### **array (entrée)**

Mémoire tampon contenant le tableau d'octets qui est lu et longueur de la mémoire tampon en octets.

Si le nombre d'octets du tableau est inférieur ou égal à la longueur de la mémoire tampon, l'ensemble du tableau est lu dans la mémoire tampon. Si le nombre d'octets du tableau est supérieur ou égal à la longueur de la mémoire tampon, celle-ci est remplie avec une partie du tableau et le curseur interne marque la position de l'octet suivant à lire. Un appel ultérieur à `readBytes()` lit les octets du tableau à partir de la position actuelle du curseur.

Si vous spécifiez un pointeur Null en entrée, l'appel ignore le tableau et ne le lit pas.

### **Retour :**

Nombre d'octets lus dans la mémoire tampon. Si la mémoire tampon est partiellement remplie, la valeur est inférieure à la longueur de la mémoire tampon, indiquant qu'il ne reste plus d'octets à lire dans le tableau. S'il ne reste plus d'octets à lire à partir du tableau avant l'appel, la valeur est `XMSC_END_OF_BYTEARRAY`.

Si vous spécifiez un pointeur Null en entrée, la méthode ne renvoie aucune valeur.

### **Exceptions :**

- `XMSEException`
- `MessageNotReadableException`
- `MessageEOFException`

## *ReadChar - Lecture de caractère*

### **Interface :**

```
Char ReadChar();
```

Lit un caractère de 2 octets depuis le flux de message.

### **Paramètres :**

Aucun

### **Retour :**

Caractère lu.

### **Exceptions :**

- `XMSEException`
- `MessageNotReadableException`
- `MessageEOFException`

## *ReadDouble - Lecture d'un nombre en virgule flottante à double précision*

### **Interface :**

```
Double ReadDouble();
```

Lit un nombre en virgule flottante à double précision à partir du flux de message.



**Paramètres :**

Aucun

**Retour :**

Le nombre en virgule flottante à double précision lu.

**Exceptions :**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadFloat - Lecture du nombre en virgule flottante*

**Interface :**

```
Single ReadFloat();
```

Lit un nombre en virgule flottante à 4 octets à partir du flux de message.

**Paramètres :**

Aucun

**Retour :**

Le nombre en virgule flottante lu.

**Exceptions :**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadInt - Lecture d'un entier*

**Interface :**

```
Int32 ReadInt();
```

Lit un entier signé de 32 bits depuis le flux de message.

**Paramètres :**

Aucun

**Retour :**

Entier lu.

**Exceptions :**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadLong - Lecture d'un entier long*

**Interface :**

```
Int64 ReadLong();
```

Lit un entier signé de 64 bits depuis le flux de message.

**Paramètres :**

Aucun

**Retour :**

Entier long lu.

**Exceptions :**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadObject - Lecture d'objet*

**Interface :**

```
Object ReadObject();
```

Lit une valeur à partir du flux de message et renvoie son type de données.

**Paramètres :**

Aucun

**Retour :**

La valeur, dont le type d'objet peut être :

Boolean  
Byte  
Byte[]  
Char  
Double  
Single  
Int32  
Int64  
Int16  
String

**Exceptions :**

XMSEException

*ReadShort - Lecture d'un entier court*

**Interface :**

```
Int16 ReadShort();
```

Lit un entier signé de 16 bits depuis le flux de message.

**Paramètres :**

Aucun

**Retour :**

Entier court lu.

**Exceptions :**

- XMSEException
- MessageNotReadableException
- MessageEOFException

### *ReadString - Lecture de chaîne*

#### **Interface :**

```
String ReadString();
```

Lit un chaîne depuis le flux de messages. Si nécessaire, XMS convertit les caractères en chaîne dans la page de codes locale.

#### **Paramètres :**

Aucun

#### **Retour :**

Objet String encapsulant la chaîne lue. Si une conversion de données est requise, il s'agit de la chaîne après conversion.

#### **Exceptions :**

- XMSEException
- MessageNotReadableException
- MessageEOFException

### *Reset - Réinitialisation*

#### **Interface :**

```
void Reset();
```

Met le corps du message en mode lecture seule et repositionne le curseur au début du flux de message.

#### **Paramètres :**

Aucun

#### **Retour :**

Vide

#### **Exceptions :**

- XMSEException
- MessageNotReadableException
- MessageEOFException

### *WriteBoolean - Ecriture d'une valeur booléenne*

#### **Interface :**

```
void WriteBoolean(Boolean value);
```

Ecrit une valeur booléenne sur un flux de message.

#### **Paramètres :**

##### **value (entrée)**

Valeur booléenne à écrire.

#### **Retour :**

Vide

#### **Exceptions :**

- XMSEException
- MessageNotWritableException

### *WriteByte - Ecriture d'un octet*

#### **Interface :**

```
void WriteByte(Byte value);  
void WriteSignedByte(Int16 value);
```

Ecrit un octet sur le flux de message.

#### **Paramètres :**

**value (entrée)**  
Octet à écrire.

#### **Retour :**

Vide

#### **Exceptions :**

- XMSEException
- MessageNotWritableException

### *WriteBytes - Ecriture d'octets*

#### **Interface :**

```
void WriteBytes(Byte[] value);
```

Ecrit un tableau d'octets sur un flux de message.

#### **Paramètres :**

**value (entrée)**  
Tableau des octets à écrire.

**length (entrée)**  
Nombre d'octets dans le tableau.

#### **Retour :**

Vide

#### **Exceptions :**

- XMSEException
- MessageNotWritableException

### *WriteChar - Ecriture d'un caractère*

#### **Interface :**

```
void WriteChar(Char value);
```

Ecrit un caractère sur le flux de message sous forme de 2 octets, l'octet de poids fort en premier.

#### **Paramètres :**

**value (entrée)**  
Caractère à écrire.

#### **Retour :**

Vide

#### **Exceptions :**

- XMSEException
- MessageNotWritableException

*WriteDouble - Ecriture d'un nombre en virgule flottante à double précision*

**Interface :**

```
void WriteDouble(Double value);
```

Convertit un nombre en virgule flottante à double précision en entier long et écrit celui-ci sur le flux de message d'octets sous forme de 8 octets, l'octet de poids fort en premier.

**Paramètres :**

**value (entrée)**

Le nombre en virgule flottante à double précision à écrire.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageNotWritableException

*WriteFloat - Ecriture d'un nombre en virgule flottante*

**Interface :**

```
void WriteFloat(Single value);
```

Convertit un nombre en virgule flottante en entier et écrit cet entier sur le flux de message sous forme de 4 octets, l'octet de poids fort en premier.

**Paramètres :**

**value (entrée)**

Le nombre en virgule flottante à écrire.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageNotWritableException

*WriteInt - Ecriture d'un entier*

**Interface :**

```
void WriteInt(Int32 value);
```

Écrit un entier sur le flux de message d'octets sous forme de 4 octets, l'octet de poids fort en premier.

**Paramètres :**

**value (entrée)**

Entier à écrire.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageNotWritableException

*WriteLong - Ecriture d'un entier long*

**Interface :**

```
void WriteLong(Int64 value);
```

Ecrit un entier long sur le flux de message sous forme de 8 octets, l'octet de poids fort en premier.

**Paramètres :**

**value (entrée)**

Entier long à écrire.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageNotWritableException

*WriteObject - Ecriture d'un objet*

**Interface :**

```
void WriteObject(Object value);
```

Ecrit une valeur, avec un type de données spécifié, sur le flux de message.

**Paramètres :**

**objectType (entrée)**

La valeur, dont le type d'objet peut être :

Boolean  
Byte  
Byte[]  
Char  
Double  
Single  
Int32  
Int64  
Int16  
String

**value (entrée)**

Tableau d'octets contenant la valeur à écrire.

**length (entrée)**

Nombre d'octets dans le tableau.

**Retour :**

Vide

**Exceptions :**

- XMSEException

*WriteShort - Ecriture d'un entier court*

**Interface :**

```
void WriteShort(Int16 value);
```

Ecrit un entier court sur le flux de message sous forme de 2 octets, l'octet de poids fort en premier.

**Paramètres :**

**value (entrée)**

Entier court à écrire.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageNotWritableException

*WriteString - Ecriture d'une chaîne*

**Interface :**

```
void WriteString(String value);
```

Ecrit une chaîne sur le flux de message.

**Paramètres :**

**value (entrée)**

Objet String encapsulant la chaîne à écrire.

**Retour :**

Vide

**Exceptions :**

- XMSEException
- MessageNotWritableException

**Propriétés et méthodes héritées**

Les propriétés suivantes sont héritées de l'interface [IMessage](#) :

[JMSCorrelationID](#), [JMSDeliveryMode](#), [JMSDestination](#), [JMSExpiration](#), [JMSMessageID](#), [JMSPriority](#), [JMSRedistribuée](#), [JMSReplyTo](#), [JMSTimestamp](#), [JMSType](#), [Propriétés](#)

Les méthodes suivantes sont héritées de l'interface [IMessage](#) :

[clearBody](#), [clearProperties](#), [PropertyExists](#)

Les méthodes suivantes sont héritées de l'interface [IPropertyContext](#) :

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

**ITextMessage**

Un message texte est un message dont le corps contient une chaîne.

**Hérarchie d'héritage :**

```
IBM.XMS.IPropertyContext
|
+---- IBM.XMS.IMessage
      |
      +---- IBM.XMS.ITextMessage
```

## **.NETpropriétés**

Text - Extraction et définition de texte

### **Interface :**

```
String Text
{
    get;
    set;
}
```

Extrait et définit la chaîne qui forme le corps du message texte.

Si nécessaire, XMS convertit les caractères en chaîne dans la page de codes locale.

### **Exceptions :**

- [XMSEException](#)
- [MessageNotReadableException](#)
- [MessageNotWritableException](#)
- [MessageEOFException](#)

### **Propriétés et méthodes héritées**

Les propriétés suivantes sont héritées de l'interface [IMessage](#) :

[JMSCorrelationID](#), [JMSDeliveryMode](#), [JMSDestination](#), [JMSExpiration](#), [JMSMessageID](#), [JMSPriority](#), [JMSRedistribuée](#), [JMSReplyTo](#), [JMSTimestamp](#), [JMSType](#), [Propriétés](#)

Les méthodes suivantes sont héritées de l'interface [IMessage](#) :

[clearBody](#), [clearProperties](#), [PropertyExists](#)

Les méthodes suivantes sont héritées de l'interface [IPropertyContext](#) :

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

## **TransactionInProgressException**

XMS émet cette exception si une application demande une opération qui n'est pas valide car une transaction est en cours.

### **Hiérarchie d'héritage :**

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
|
+----IBM.XMS.TransactionInProgressException
```

### **Propriétés et méthodes héritées**

Les méthodes suivantes sont héritées de l'interface [XMSEException](#) :

[GetErrorCode](#), [GetLinkedException](#)



## TransactionRolledBackException

XMS émet cette exception si une application appelle `Session.commit()` pour valider la transaction en cours, mais que la transaction est ensuite annulée.

### Hiérarchie d'héritage :

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
|
+----IBM.XMS.TransactionRolledBackException
```

### Propriétés et méthodes héritées

Les méthodes suivantes sont héritées de l'interface [XMSEException](#) :

[GetErrorCode](#), [GetLinkedException](#)

## XMSEException

Si XMS détecte une erreur lors du traitement d'un appel à une méthode .NET , XMS émet une exception. Une exception est un objet qui encapsule des informations relatives à l'erreur.

### Hiérarchie d'héritage :

```
System.Exception
|
+----IBM.XMS.XMSEException
```

Il existe différents types d'exception XMS et un objet `XMSEException` n'est qu'un seul type d'exception. Toutefois, la classe `XMSEException` est une superclasse des classes d'exception XMS. XMS émet un objet `XMSEException` si aucun autre type d'exception n'est approprié.

### .NETpropriétés

*ErrorCode* - Extraction du code d'erreur

#### Interface :

```
public String ErrorCode
{
    get {return errorCode_;}
}
```

Extrait le code d'erreur.

#### Exceptions :

- `XMSEException`

*LinkedException* - Extraction de l'exception associée

#### Interface :

```
public Exception LinkedException
{
    get { return linkedException_;}
    set { linkedException_ = value;}
}
```

Extrait l'exception suivante dans la chaîne d'exceptions.

La méthode renvoie une valeur `Null` s'il n'y a plus d'exception dans la chaîne.

**Exceptions :**

- XMSEException

**XMSFactoryFactory**

Si une application n'utilise pas les objets gérés, utilisez cette classe pour créer des fabriques de connexions, des files d'attente et des rubriques.

**Hiérarchie d'héritage :**

Aucun

**.NETpropriétés**

*Metadata - Extraction de métadonnées*

**Interface :**

```
IConnectionMetaData MetaData
```

Extrait les métadonnées appropriées pour le type de connexion de l'objet XMSFactoryFactory.

**Exceptions :**

Aucun

**Méthodes**

*CreateConnectionFactory - Création d'une fabrique de connexions*

**Interface :**

```
IConnectionFactory CreateConnectionFactory();
```

Crée un objet ConnectionFactory du type déclaré.

**Paramètres :**

Aucun

**Retour :**

Objet ConnectionFactory.

**Exceptions :**

- XMSEException

*CreateQueue - Création d'une file d'attente*

**Interface :**

```
IDestination CreateQueue(String name);
```

Crée un objet Destination pour représenter la file d'attente sur le serveur de messagerie.

Cette méthode ne permet pas de créer la file d'attente sur le serveur de messagerie. Vous devez créer celle-ci avant qu'une application puisse appeler cette méthode.

**Paramètres :****name (entrée)**

Un objet de type String encapsulant le nom de la file d'attente ou un identificateur URI (Uniform Resource Identifier) qui identifie la file d'attente.

**Retour :**

Objet Destination représentant la file d'attente.

**Exceptions :**

- XMSEException

*CreateTopic - Création d'une rubrique*

**Interface :**

```
IDestination CreateTopic(String name);
```

Crée un objet Destination pour représenter une rubrique.

**Paramètres :****name (entrée)**

Objet String encapsulant le nom de la rubrique ou un identificateur URI qui identifie la rubrique.

**Retour :**

Objet Destination représentant la rubrique.

**Exceptions :**

- XMSEException

*GetInstance - Extraction d'une instance de XMSFactoryFactory*

**Interface :**

```
static XMSFactoryFactory GetInstance(int connectionType);
```

Créez une instance de XMSFactoryFactory. Une application XMS utilise un objet XMSFactoryFactory pour obtenir une référence à un objet ConnectionFactory approprié pour le type de protocole requis. Cet objet ConnectionFactory peut ensuite produire des connexions uniquement pour ce type de protocole.

**Paramètres :****connectionType (entrée)**

Le type de connexion pour laquelle l'objet ConnectionFactory génère des connexions :

- XMSC.CT\_WPM
- XMSC.CT\_RTT
- XMSC.CT\_WMQ

**Retour :**

Objet XMSFactoryFactory dédié au type de connexion déclaré.

**Exceptions :**

- NotSupportedException

## Propriétés des objets XMS

Cette section décrit les propriétés d'objet définies par XMS.

Cette section contient des informations sur les types d'objet suivants:

- [«Propriétés de Connection»](#), à la page 2124
- [«Propriétés de ConnectionFactory»](#), à la page 2125
- [«Propriétés de ConnectionMetaData»](#), à la page 2129
- [«Propriétés de Destination»](#), à la page 2130
- [«Propriétés de InitialContext»](#), à la page 2131
- [«Propriétés de Message»](#), à la page 2132
- [«Propriétés de MessageConsumer»](#), à la page 2137

- «Propriétés de MessageProducer», à la page 2138
- «Propriétés de Session», à la page 2138

La description de chaque type d'objet répertorie les propriétés d'un objet du type spécifié et fournit une brève description de chaque propriété.

Cette section fournit également une définition de chaque propriété (voir «Définitions de propriété», à la page 2138).

Si une application définit ses propres propriétés des objets décrits dans cette section, elle ne génère pas d'erreur, mais peut entraîner des résultats imprévisibles.

**Remarque :** Les noms de propriété et les valeurs de cette section sont affichés sous la forme `XMSC.NAME`, qui est la forme utilisée pour C et C++. Toutefois, dans .NET, la forme du nom de propriété peut être `XMSC.NAME` ou `XMSC_NAME`, selon la façon dont vous l'utilisez:

- Si vous spécifiez une propriété, le nom de la propriété doit être au format `XMSC.NAME`, comme illustré dans l'exemple suivant:

```
cf.SetStringProperty(XMSC.WMQ_CHANNEL, "DOTNET.SVRCONN");
```

- Si vous spécifiez une chaîne, le nom de la propriété doit être au format `XMSC_NAME`, comme illustré dans l'exemple suivant:

```
cf.SetStringProperty("XMSC_WMQ_CHANNEL", "DOTNET.SVRCONN");
```

Dans .NET, les noms et les valeurs de propriété sont fournis en tant que constantes dans la classe `XMSC`. Ces constantes identifient des chaînes et sont utilisées par n'importe quelle application `XMS.NET`. Si vous utilisez ces constantes prédéfinies, les noms et les valeurs de propriété sont au format `XMSC.NOM`; ainsi, vous utiliserez `XMSC.USERID` plutôt que `XMSC_USERID`.

Les types de données sont également au format utilisé pour C/C++. Vous trouverez les valeurs correspondantes pour .NET dans [Types de données pour .NET](#).

## Propriétés de Connection

Présentation des propriétés de l'objet `Connection`, avec des liens vers des informations de référence plus détaillées.

Nom de la propriété	Description
<a href="#">«XMSC_WMQ_RESOLVED_QUEUE_MANAGER»</a> , à la page 2173	Cette propriété est utilisée pour obtenir le nom du gestionnaire de files d'attente auquel il est connecté.
<a href="#">«XMSC_WMQ_RESOLVED_QUEUE_MANAGER_ID»</a> , à la page 2173	Cette propriété est remplie avec l'ID du gestionnaire de files d'attente après la connexion.
<a href="#">XMSC_WPM_CONNECTION_PROTOCOL</a>	Protocole de communication utilisé pour la connexion au moteur de messagerie. Cette propriété est en lecture seule.
<a href="#">XMSC_WPM_HOST_NAME</a>	Nom hôte ou adresse IP du système qui contient le moteur de messagerie auquel l'application est connectée. Cette propriété est en lecture seule.
<a href="#">XMSC_WPM_ME_NAME</a>	Nom du moteur de messagerie auquel l'application est connectée. Cette propriété est en lecture seule.
<a href="#">XMSC_WPM_PORT</a>	Numéro du port d'écoute du moteur de messagerie auquel l'application est connectée. Cette propriété est en lecture seule.

Un objet `Connection` a également des propriétés en lecture seule dérivées des propriétés de la fabrique de connexions qui a été utilisée pour créer la connexion. Ces propriétés sont dérivées non seulement

des propriétés de fabrique de connexions définies lors de la création de la connexion, mais aussi des valeurs par défaut des propriétés qui n'ont pas été définies. Les propriétés incluent uniquement celles qui sont pertinentes pour le type de serveur de messagerie auquel l'application est connectée. Les noms de propriété sont les mêmes que les noms des propriétés de fabrique de connexions.

## Propriétés de ConnectionFactory

Présentation des propriétés de l'objet ConnectionFactory, avec des liens vers des informations de référence plus détaillées.

Nom de la propriété	Description
«XMSC_ASYNC_EXCEPTIONS», à la page 2148	Cette propriété détermine si XMS informe un programme d'écoute des exceptions uniquement lorsqu'une connexion est interrompue, ou également lorsqu'une exception est émise de façon asynchrone sur un appel d'API XMS. Cette propriété s'applique à toutes les connexions créées à partir de la fabrique de connexions et pour lesquelles un programme d'écoute des exceptions est enregistré.
<b>V 9.3.0</b> «XMSC_WMQ_BALANCING_APPLICATION_TYPE», à la page 2157	Type d'option d'équilibrage
<b>V 9.3.0</b> «XMSC_WMQ_BALANCING_OPTIONS», à la page 2157	Options d'équilibrage définies par l'application émettrice
<b>V 9.3.0</b> «XMSC_WMQ_BALANCING_TIMEOUT», à la page 2158	Délai après lequel le rééquilibrage peut interrompre l'activité de l'application.
<u>XMSC_CLIENT_ID</u>	Identificateur de client pour une connexion.
<u>XMSC_CONNECTION_TYPE</u>	Type de serveur de messagerie auquel une application se connecte.
<u>XMSC_PASSWORD</u>	Mot de passe pouvant être utilisé pour authentifier l'application lorsqu'elle tente de se connecter à un serveur de messagerie.
«XMSC_RTT_BROKER_PING_INTERVAL», à la page 2153	Intervalle de temps, en millisecondes, après lequel XMS.NET vérifie la connexion à un serveur de messagerie en temps réel pour détecter une activité.
<u>XMSC_RTT_CONNECTION_PROTOCOL</u>	Protocole de communication utilisé pour une connexion en temps réel à un courtier.
<u>XMSC_RTT_HOST_NAME</u>	Nom d'hôte ou adresse IP du système sur lequel un courtier s'exécute.
<u>XMSC_RTT_LOCAL_ADDRESS</u>	Nom d'hôte ou adresse IP de l'interface de réseau local à utiliser pour une connexion en temps réel à un courtier.
<u>XMSC_RTT_MULTICAST</u>	Paramètre de multidiffusion pour une fabrique de connexions ou une destination.
<u>XMSC_RTT_PORT</u>	Numéro du port sur lequel un courtier écoute les demandes entrantes.

Tableau 873. Propriétés de ConnectionFactory (suite)

Nom de la propriété	Description
<u>XMSC_USERID</u>	ID utilisateur pouvant être utilisé pour authentifier l'application lorsqu'elle tente de se connecter à un serveur de messagerie.
<u>XMSC_WMQ_BROKER_CONTROLQ</u>	Nom de la file d'attente de contrôle utilisée par un courtier.
<u>XMSC_WMQ_BROKER_PUBQ</u>	Nom de la file d'attente contrôlée par un courtier lorsque des applications envoient les messages qu'elles publient.
<u>XMSC_WMQ_BROKER_QMGR</u>	Nom du gestionnaire de files d'attente auquel un courtier est connecté.
<u>XMSC_WMQ_BROKER_SUBQ</u>	Nom de la file d'attente de souscription pour un consommateur de message non durable.
<u>XMSC_WMQ_BROKER_VERSION</u>	Type de courtier utilisé par l'application pour une connexion ou pour la destination.
« <u>XMSC_WMQ_CCDTURL</u> », à la page 2160	Adresse URL qui identifie le nom et l'emplacement du fichier contenant la table de définition de canal du client et qui spécifie le mode d'accès au fichier.
<u>XMSC_WMQ_CHANNEL</u>	Nom du canal à utiliser pour une connexion.
« <u>XMSC_WMQ_CLIENT_RECONNECT_OPTIONS</u> », à la page 2161	Cette propriété spécifie les options de reconnexion client pour les nouvelles connexions créées par cette fabrique
« <u>XMSC_WMQ_CLIENT_RECONNECT_TIMEOUT</u> », à la page 2161	Cette propriété spécifie le temps, en secondes, qu'un connexion client utilise pour essayer de se reconnecter.
<u>XMSC_WMQ_CONNECTION_MODE</u>	Mode par lequel une application se connecte à un gestionnaire de files d'attente.
« <u>XMSC_WMQ_CONNECTION_NAME_LIST</u> », à la page 2162	Cette propriété spécifie les hôtes auxquels le client tente de se reconnecter après une interruption de sa connexion.
<u>XMSC_WMQ_FAIL_IF QUIESCE</u>	Indique que les appels à certaines méthodes échouent si le gestionnaire de files d'attente auquel l'application est connectée est à l'état de mise au repos.
<u>XMSC_WMQ_HOST_NAME</u>	Nom d'hôte ou adresse IP du système sur lequel s'exécute un gestionnaire de files d'attente.
<u>XMSC_WMQ_LOCAL_ADDRESS</u>	Pour une connexion à un gestionnaire de files d'attente, cette propriété spécifie l'interface de réseau local et/ou le port ou la plage de ports à utiliser.
<u>XMSC_WMQ_MESSAGE_SELECTION</u>	Détermine si la sélection des messages est effectuée par le client XMS ou par le courtier.
<u>XMSC_WMQ_MSG_BATCH_SIZE</u>	Nombre maximal de messages à extraire d'une file d'attente en un seul lot lorsque la distribution asynchrone de messages est utilisée.
<u>XMSC_WMQ_POLLING_INTERVAL</u>	Si chaque programme d'écoute de message dans une session ne comporte pas de message approprié dans sa file d'attente, cette valeur est l'intervalle de temps maximal, en millisecondes, qui s'écoule avant que chaque programme d'écoute de message essaie à nouveau d'obtenir un message à partir de sa file d'attente.

Tableau 873. Propriétés de ConnectionFactory (suite)

Nom de la propriété	Description
«XMSC_WMQ_PROVIDER_VERSION», à la page <a href="#">2171</a>	Version, édition, niveau de modification et groupe de correctifs du gestionnaire de files d'attente auquel l'application a l'intention de se connecter.
<a href="#">XMSC_WMQ_PORT</a>	Numéro du port sur lequel un gestionnaire de files d'attente écoute les demandes entrantes.
<a href="#">XMSC_WMQ_PUB_ACK_INTERVAL</a>	Nombre de messages publiés par un diffuseur de publications avant que le client XMS ne demande un accusé de réception du courtier.
«XMSC_WMQ_PUT_ASYNC_ALLOWED», à la page <a href="#">2166</a>	Cette propriété détermine si les expéditeurs de message sont autorisés à utiliser les insertions asynchrones pour envoyer des messages à cette destination.
<a href="#">XMSC_WMQ_QMGR_CCSD</a>	Identificateur (CCSID) du jeu de caractères codés, ou page de codes, dans lequel les zones de données de type caractères définies dans l'interface MQI (Message Queue Interface) sont échangées entre le client XMS et le client IBM MQ .
<a href="#">XMSC_WMQ_QUEUE_MANAGER</a>	Nom du gestionnaire de files d'attente auquel se connecter.
<a href="#">XMSC_WMQ_RECEIVE_EXIT</a>	Identifie un exit de réception de canal à exécuter.
<a href="#">XMSC_WMQ_RECEIVE_EXIT_INIT</a>	Données utilisateur transmises à un exit de réception de canal lors de l'appel.
<a href="#">XMSC_WMQ_SECURITY_EXIT</a>	Identifie un exit de sécurité de canal.
<a href="#">XMSC_WMQ_SECURITY_EXIT_INIT</a>	Données utilisateur transmises à un exit de sécurité de canal lors de l'appel.
«XMSC_WMQ_SEND_CHECK_COUNT», à la page <a href="#">2175</a>	Nombre d'appels d'envoi à autoriser entre deux vérifications d'erreurs asynchrones, au cours d'une même session XMS non transactionnelle.
<a href="#">XMSC_WMQ_SEND_EXIT</a>	Identifie un exit d'émission de canal.
<a href="#">XMSC_WMQ_SEND_EXIT_INIT</a>	Données utilisateur qui sont transmises aux exits d'émission une fois ceux-ci appelés.
«XMSC_WMQ_SHARE_CONV_ALLOWED», à la page <a href="#">2175</a>	Indique si une connexion client peut partager son socket avec d'autres connexions XMS de niveau supérieur à partir du même processus vers le même gestionnaire de files d'attente, si les définitions de canal correspondent. Cette propriété est fournie pour permettre un isolement complet des connexions dans des sockets distincts, lorsque cela est requis pour des raisons de développement, de maintenance ou de fonctionnement.
<a href="#">XMSC_WMQ_SSL_CERT_STORES</a>	Emplacements des serveurs contenant les listes de révocation de certificat à utiliser sur une connexion SSL à un gestionnaire de files d'attente.
<a href="#">XMSC_WMQ_SSL_CIPHER_SPEC</a>	Nom de la spécification CipherSpec à utiliser sur une connexion sécurisée vers un gestionnaire de files d'attente.

<i>Tableau 873. Propriétés de ConnectionFactory (suite)</i>	
<b>Nom de la propriété</b>	<b>Description</b>
<u><a href="#">XMSC_WMQ_SSL_CIPHER_SUITE</a></u>	Nom de la suite de chiffrement à utiliser sur une connexion TLS à un gestionnaire de files d'attente. Le protocole utilisé lors de la négociation de la connexion sécurisée dépend de la suite de chiffrement spécifiée.
<u><a href="#">XMSC_WMQ_SSL_CRYPT_HW</a></u>	Détails de configuration pour le matériel de cryptographie connecté au système client.
<u><a href="#">XMSC_WMQ_SSL_FIPS_REQUIRED</a></u>	La valeur de cette propriété détermine si une application peut ou non utiliser des suites de chiffrement conformes non FIPS. Si cette propriété est pour valeur true, seuls les algorithmes FIPS sont utilisés pour la connexion client/serveur.
<u><a href="#">XMSC_WMQ_SSL_KEY_REPOSITORY</a></u>	Emplacement du fichier de clés dans lequel les clés et les certificats sont stockés.
<u><a href="#">XMSC_WMQ_SSL_KEY_RESETCOUNT</a></u>	Nombre total d'octets non chiffrés envoyés et reçus dans une conversation SSL avant renégociation de la clé confidentielle.
<u><a href="#">XMSC_WMQ_SSL_PEER_NAME</a></u>	Nom d'homologue à utiliser sur une connexion SSL vers un gestionnaire de files d'attente.
<u><a href="#">XMSC_WMQ_SYNCPOINT_ALL_GETS</a></u>	Indique si tous les messages doivent être extraits des files d'attente sous le contrôle d'un point de synchronisation.
« <u><a href="#">XMSC_WMQ_TARGET_CLIENT</a></u> », à la page 2182	
<u><a href="#">XMSC_WMQ_TEMP_Q_PREFIX</a></u>	Préfixe utilisé pour former le nom de la file d'attente dynamique IBM MQ créée lorsque l'application crée une file d'attente temporaire XMS .
<u><a href="#">XMSC_WMQ_TEMP_TOPIC_PREFIX</a></u>	Lors de la création de rubriques temporaires, XMS génère une chaîne de rubrique au format "TEMP/TEMPTOPICPREFIX/unique_id", ou si cette propriété contient la valeur par défaut, cette chaîne, "TEMP/unique_id", est générée. La spécification d'une valeur non vide permet la définition de files d'attente modèle spécifiques pour la création des files d'attente gérées pour les abonnés à des rubriques temporaires créées sous cette connexion.
<u><a href="#">XMSC_WMQ_TEMPORARY_MODEL</a></u>	Nom de la file d'attente modèle IBM MQ à partir de laquelle une file d'attente dynamique est créée lorsque l'application crée une file d'attente temporaire XMS .
<u><a href="#">XMSC_WPM_BUS_NAME</a></u>	Pour une fabrique de connexions, nom du bus d'intégration de services auquel l'application se connecte ; pour une destination, nom du bus d'intégration de services dans lequel la destination existe.
<u><a href="#">XMSC_WPM_CONNECTION_PROXIMITY</a></u>	Paramètre de proximité de connexion pour la connexion.
<u><a href="#">XMSC_WPM_DUR_SUB_HOME</a></u>	Nom du moteur de messagerie où sont gérés les abonnements durables pour une connexion ou une destination.



Tableau 873. Propriétés de ConnectionFactory (suite)

Nom de la propriété	Description
<a href="#">XMSC_WPM_LOCAL_ADDRESS</a>	Pour une connexion à un bus d'intégration de services, cette propriété spécifie l'interface de réseau local et/ou le port ou la plage de ports à utiliser.
<a href="#">XMSC_WPM_NON_PERSISTENT_MAP</a>	Niveau de fiabilité des messages non persistants envoyés à l'aide de la connexion.
<a href="#">XMSC_WPM_PERSISTENT_MAP</a>	Niveau de fiabilité des messages non persistants envoyés à l'aide de la connexion.
<a href="#">XMSC_WPM_PROVIDER_ENDPOINTS</a>	Séquence d'une ou plusieurs adresses de noeud final de serveurs d'amorçage.
<a href="#">XMSC_WPM_TARGET_GROUP</a>	Nom d'un groupe cible de moteurs de messagerie.
<a href="#">XMSC_WPM_TARGET_SIGNIFICANCE</a>	Signification du groupe cible de moteurs de messagerie.
<a href="#">XMSC_WPM_TARGET_TRANSPORT_CHAIN</a>	Nom de la chaîne de transport entrant que l'application doit utiliser pour se connecter à un moteur de messagerie.
<a href="#">XMSC_WPM_TARGET_TYPE</a>	Type du groupe cible de moteurs de messagerie.
<a href="#">XMSC_WPM_TEMP_Q_PREFIX</a>	Préfixe utilisé pour former le nom de la file d'attente temporaire créée dans le bus d'intégration de services lorsque l'application crée une file d'attente temporaire XMS .
<a href="#">XMSC_WPM_TEMP_TOPIC_PREFIX</a>	Préfixe utilisé pour former le nom d'une rubrique temporaire créée par l'application.

## Propriétés de ConnectionMetaData

Présentation des propriétés de l'objet ConnectionMetaData, avec des liens vers des informations de référence plus détaillées.

Tableau 874. Propriétés de ConnectionMetaData

Nom de la propriété	Description
<a href="#">XMSC_JMS_MAJOR_VERSION</a>	Numéro de version principale de la spécification JMS sur laquelle repose XMS . Cette propriété est en lecture seule.
<a href="#">XMSC_JMS_MINOR_VERSION</a>	Numéro de version secondaire de la spécification JMS sur laquelle repose XMS . Cette propriété est en lecture seule.
<a href="#">XMSC_JMS_VERSION</a>	Identificateur de version de la spécification JMS sur laquelle est basée XMS . Cette propriété est en lecture seule.
<a href="#">XMSC_MAJOR_VERSION</a>	Numéro de version du client XMS . Cette propriété est en lecture seule.
<a href="#">XMSC_MINOR_VERSION</a>	Numéro d'édition du client XMS . Cette propriété est en lecture seule.
<a href="#">XMSC_PROVIDER_NAME</a>	Fournisseur du client XMS . Cette propriété est en lecture seule.
<a href="#">XMSC_VERSION</a>	Identificateur de version de l'interface de ligne de commandeXMSent. Cette propriété est en lecture seule.

## Propriétés de Destination

Présentation des propriétés de l'objet Destination, avec des liens vers des informations de référence plus détaillées.

<i>Tableau 875. Propriétés de Destination</i>	
Nom de la propriété	Description
<a href="#">XMSC_DELIVERY_MODE</a>	Mode de livraison des messages envoyés à la destination.
<a href="#">XMSC_PRIORITY</a>	Priorité des messages envoyés à la destination.
<a href="#">XMSC_RTT_MULTICAST</a>	Paramètre de multidiffusion pour une fabrique de connexions ou une destination.
<a href="#">XMSC_TIME_TO_LIVE</a>	Durée de vie des messages envoyés à la destination.
<a href="#">XMSC_WMQ_BROKER_VERSION</a>	Type de courtier utilisé par l'application pour une connexion ou pour la destination.
<a href="#">XMSC_WMQ_CCSID</a>	Identificateur (CCSID) du jeu de caractères codés, ou page de codes, dans lequel se trouvent les chaînes de données de type caractères dans le corps d'un message lorsque le client XMS transmet le message à la destination.
<a href="#">XMSC_WMQ_DUR_SUBQ</a>	Nom de la file d'attente de souscription pour un abonné durable qui reçoit des messages de la destination.  <b>Remarque :</b> Cette propriété peut être utilisée avec la version 2.0 de IBM Message Service Client for .NET mais n'a aucun effet sur une application connectée à un gestionnaire de files d'attente IBM WebSphere MQ 7.0 , sauf si la propriété XMSC_WMQ_PROVIDER_VERSION de la fabrique de connexions est définie sur un numéro de version inférieur à 7.
<a href="#">XMSC_WMQ_ENCODING</a>	Mode de représentation des données numériques dans le corps d'un message lorsque le client XMS transmet le message à la destination.
<a href="#">XMSC_WMQ_FAIL_IF_QUIESCE</a>	Indique que les appels à certaines méthodes échouent si le gestionnaire de files d'attente auquel l'application est connectée est à l'état de mise au repos.
« <a href="#">XMSC_WMQ_MESSAGE_BODY</a> », à la page 2164	Cette propriété détermine si une application XMS traite l' MQRFH2 d'un message IBM MQ dans le cadre de la charge utile du message (c'est-à-dire dans le cadre du corps du message).
« <a href="#">XMSC_WMQ_MQMD_MESSAGE_CONTEXT</a> », à la page 2165	Détermine le niveau de contexte de message à définir par l'application XMS . L'application doit être exécutée avec les droits appropriés pour que cette propriété soit appliquée.
« <a href="#">XMSC_WMQ_MQMD_READ_ENABLED</a> », à la page 2165	Cette propriété détermine si une application XMS peut extraire les valeurs des zones MQMD ou non.
« <a href="#">XMSC_WMQ_MQMD_WRITE_ENABLED</a> », à la page 2166	Cette propriété détermine si une application XMS peut ou non définir les valeurs des zones MQMD.

<i>Tableau 875. Propriétés de Destination (suite)</i>	
<b>Nom de la propriété</b>	<b>Description</b>
<u>«XMSC_WMQ_READ_AHEAD_ALLOWED»</u> , à la page 2167	Cette propriété détermine si les consommateurs de message et les navigateurs de files d'attente sont autorisés à utiliser la lecture anticipée pour extraire les messages non permanents et non transactionnels de cette destination dans une mémoire tampon interne avant de les recevoir.
<u>«XMSC_WMQ_READ_AHEAD_CLOSE_POLICY»</u> , à la page 2167	Cette propriété détermine, pour les messages distribués à un programme d'écoute de messages asynchrone, ce qu'il advient des messages dans la mémoire tampon interne de lecture anticipée lorsque le consommateur de messages est fermé.
<u>«XMSC_WMQ_RECEIVE_CCSD»</u> , à la page 2172	Propriété de destination qui définit le CCSID cible pour la conversion des messages du gestionnaire de files d'attente. La valeur est ignorée sauf si XMSC_WMQ_RECEIVE_CONVERSION est défini sur WMQ_RECEIVE_CONVERSION_QMGR.
<u>«XMSC_WMQ_RECEIVE_CONVERSION»</u> , à la page 2172	Propriété de destination qui détermine si la conversion de données sera effectuée par le gestionnaire de files d'attente.
<u>XMSC_WMQ_TARGET_CLIENT</u>	Indique si les messages envoyés à la destination contiennent un en-tête MQRFH2.
<u>XMSC_WMQ_TEMP_TOPIC_PREFIX</u>	Lors de la création de rubriques temporaires, XMS génère une chaîne de rubrique au format "TEMP/TEMPTOPICPREFIX/unique_id", ou si cette propriété contient la valeur par défaut, cette chaîne, "TEMP/unique_id", est générée. La spécification d'une valeur non vide permet la définition de files d'attente modèle spécifiques pour la création des files d'attente gérées pour les abonnés à des rubriques temporaires créées sous cette connexion.
<u>XMSC_WPM_BUS_NAME</u>	Pour une fabrique de connexions, nom du bus d'intégration de services auquel l'application se connecte ; pour une destination, nom du bus d'intégration de services dans lequel la destination existe.
<u>XMSC_WPM_TOPIC_SPACE</u>	Nom de l'espace de sujet contenant la rubrique.

## Propriétés de InitialContext

Présentation des propriétés de l'objet InitialContext, avec des liens vers des informations de référence plus détaillées.

<i>Tableau 876. Propriétés de InitialContext</i>	
<b>Nom de la propriété</b>	<b>Description</b>
<u>XMSC_IC_PROVIDER_URL</u>	Utilisé pour localiser le répertoire de désignation JNDI de sorte que le service de dénomination COS n'ait pas besoin d'être sur le même serveur que le service Web.

<i>Tableau 876. Propriétés de InitialContext (suite)</i>	
<b>Nom de la propriété</b>	<b>Description</b>
<a href="#"><u>XMSC_IC_SECURITY_AUTHENTICATION</u></a>	Basé sur l'interface de contexte Java SECURITY_AUTHENTICATION. Cette propriété est applicable uniquement au contexte d'affectation de classe de service.
<a href="#"><u>XMSC_IC_SECURITY_CREDENTIALS</u></a>	Basé sur l'interface de contexte Java SECURITY_CREDENTIALS. Cette propriété est applicable uniquement au contexte d'affectation de classe de service.
<a href="#"><u>XMSC_IC_SECURITY_PRINCIPAL</u></a>	Basé sur l'interface de contexte Java SECURITY_PRINCIPAL. Cette propriété est applicable uniquement au contexte d'affectation de classe de service.
<a href="#"><u>XMSC_IC_SECURITY_PROTOCOL</u></a>	Basé sur l'interface de contexte Java SECURITY_PROTOCOL Cette propriété est applicable uniquement au contexte d'affectation de classe de service.
<a href="#"><u>XMSC_IC_URL</u></a>	Pour les contextes LDAP et FileSystem, adresse du référentiel contenant les objets gérés. Pour les contextes d'affectation de nom de classe de service, adresse du service Web qui recherche les objets dans le répertoire.

## Propriétés de Message

Présentation des propriétés de l'objet Message, avec des liens vers des informations de référence plus détaillées.

<i>Tableau 877. Propriétés de Message</i>	
<b>Nom de la propriété</b>	<b>Description</b>
<a href="#"><u>JMS_IBM_CHARACTER_SET</u></a>	Identificateur (CCSID) du jeu de caractères codés, ou page de codes, dans lequel se trouvent les chaînes de données de caractères dans le corps du message lorsque le client XMS transmet le message à la destination prévue. Dans XMS, cette propriété possède une valeur numérique et des mappes sur CCSID. Toutefois, cette propriété est basée sur une propriété JMS qui contient une valeur de type chaîne et mappe vers le jeu de caractères Java qui représente ce CCSID numérique.
<a href="#"><u>JMS_IBM_ENCODING</u></a>	Mode de représentation des données numériques dans le corps du message lorsque le client XMS transmet le message à sa destination prévue.
<a href="#"><u>JMS_IBM_EXCEPTIONMESSAGE</u></a>	Texte décrivant la raison pour laquelle le message a été envoyé à la destination d'exception. Cette propriété est en lecture seule.
<a href="#"><u>JMS_IBM_EXCEPTIONPROBLEMDESTINATION</u></a>	Nom de la destination du message avant que celui-ci soit envoyé à la destination d'exception.
<a href="#"><u>JMS_IBM_EXCEPTIONREASON</u></a>	Code anomalie indiquant la raison pour laquelle le message a été envoyé à la destination d'exception.
<a href="#"><u>JMS_IBM_EXCEPTIONTIMESTAMP</u></a>	Heure à laquelle le message a été envoyé à la destination d'exception.
<a href="#"><u>JMS_IBM_FEEDBACK</u></a>	Code qui indique la nature d'un message de rapport.

<i>Tableau 877. Propriétés de Message (suite)</i>	
<b>Nom de la propriété</b>	<b>Description</b>
<u>JMS_IBM_FORMAT</u>	Nature des données d'application dans le message.
<u>JMS_IBM_LAST_MSG_IN_GROUP</u>	Indique si le message est le dernier d'un groupe de messages.
<u>JMS_IBM_MSGTYPE</u>	Type du message.
<u>JMS_IBM_PUTAPPLTYPE</u>	Type de l'application ayant envoyé le message.
<u>JMS_IBM_PUTDATE</u>	Date d'envoi du message.
<u>JMS_IBM_PUTTIME</u>	Heure d'envoi du message.
<u>JMS_IBM_REPORT_COA</u>	Demande de messages de rapport "confirmation à l'arrivée", avec indication de la quantité de données d'application du message d'origine devant être incluse dans un message de rapport.
<u>JMS_IBM_REPORT_COD</u>	Demande de messages de rapport "confirmation à la livraison", avec indication de la quantité de données d'application du message d'origine devant être incluse dans un message de rapport.
<u>JMS_IBM_REPORT_DISCARD_MSG</u>	Demande de destruction du message au cas où celui-ci ne peut pas être livré à la destination prévue.
<u>JMS_IBM_REPORT_EXCEPTION</u>	Demande de messages de rapport d'exception, avec indication de la quantité de données d'application du message d'origine devant être incluse dans un message de rapport.
<u>JMS_IBM_REPORT_EXPIRATION</u>	Demande de messages de rapport d'expiration, avec indication de la quantité de données d'application du message d'origine devant être incluse dans un message de rapport.
<u>JMS_IBM_REPORT_NAN</u>	Demande de messages de rapport de notification d'action négative.
<u>JMS_IBM_REPORT_PAN</u>	Demande de messages de rapport de notification d'action positive.
<u>JMS_IBM_REPORT_PASS_CORREL_ID</u>	Demande que l'identificateur de corrélation de tout message de rapport ou de réponse soit le même que celui du message d'origine.
<u>JMS_IBM_REPORT_PASS_MSG_ID</u>	Demande que l'identificateur de message de tout message de rapport ou de réponse soit le même que celui du message d'origine.
<u>JMS_IBM_RETAIN</u>	La définition de cette propriété indique que le gestionnaire de file d'attente traite un message en tant que publication conservée.
<u>JMS_IBM_SYSTEM_MESSAGEID</u>	Identificateur unique du message dans le bus d'intégration de services. Cette propriété est en lecture seule.
<u>JMSX_APPID</u>	Nom de l'application ayant envoyé le message.
<u>JMSX_DELIVERY_COUNT</u>	Nombre de tentatives de livraison du message.

Tableau 877. Propriétés de Message (suite)	
Nom de la propriété	Description
<u>JMSX_GROUPID</u>	Identificateur du groupe de messages auquel le message appartient.
<u>JMSX_GROUPSEQ</u>	Numéro de séquence du message dans un groupe de messages.
<u>JMSX_USERID</u>	ID utilisateur associé à l'application ayant envoyé le message.

### Propriétés JMS\_IBM\_MQMD\*

IBM Message Service Client for .NET permet aux applications client de lire/écrire des zones MQMD à l'aide des API. Il permet également d'accéder aux données de message MQ. Par défaut, l'accès à MQMD est désactivé et doit être activé explicitement par l'application à l'aide des propriétés Destination XMSC\_WMQ\_MQMD\_WRITE\_ENABLED et XMSC\_WMQ\_MQMD\_READ\_ENABLED. Ces deux propriétés sont indépendantes l'une de l'autre.

Toutes les zones MQMD à l'exception de StrucId et Version sont exposées en tant que propriétés supplémentaires de l'objet Message et portent le préfixe JMS\_IBM\_MQMD.

Les propriétés JMS\_IBM\_MQMD\* ont priorité sur les autres propriétés JMS\_IBM\* décrites dans le tableau précédent.

### Envoi de messages

Toutes les zones MQMD à l'exception de StrucId et Version sont représentées. Ces propriétés font référence uniquement aux zones MQMD ; lorsqu'une propriété se trouve à la fois dans le MQMD et dans l'en-tête MQRFH2, la version de MQRFH2 n'est ni définie ni extraite. Toutes ces propriétés peuvent être définies, à l'exception de JMS\_IBM\_MQMD\_BackoutCount. Toute valeur définie JMS\_IBM\_MQMD\_BackoutCount est ignorée.

Si une propriété a une longueur maximale et que vous indiquez une valeur trop longue, celle-ci est tronquée.

Pour certaines propriétés, vous devez également définir la propriété XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT sur l'objet Destination. L'application doit être exécutée avec les droits appropriés pour que cette propriété soit appliquée. Si vous définissez pas une valeur appropriée pour XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT, celle-ci est ignorée. Si vous définissez une valeur appropriée pour XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT, mais que vous ne disposez pas des droits requis pour le gestionnaire de files d'attente, une exception est émise. Les propriétés qui nécessitent des valeurs spécifiques de XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT sont les suivantes.

Les propriétés suivantes requièrent que XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT soit défini sur XMSC\_WMQ\_MDCTX\_SET\_IDENTITY\_CONTEXT ou XMSC\_WMQ\_MDCTX\_SET\_ALL\_CONTEXT :

- JMS\_IBM\_MQMD\_UserIdentifier
- JMS\_IBM\_MQMD\_AccountingToken
- JMS\_IBM\_MQMD\_ApplIdentityData

Les propriétés suivantes requièrent que XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT soit défini sur XMSC\_WMQ\_MDCTX\_SET\_ALL\_CONTEXT :

- JMS\_IBM\_MQMD\_PutApplType
- JMS\_IBM\_MQMD\_PutApplName
- JMS\_IBM\_MQMD\_PutDate
- JMS\_IBM\_MQMD\_PutTime
- JMS\_IBM\_MQMD\_ApplOriginData

## Réception de messages

Toutes ces propriétés sont disponibles sur un message reçu si la propriété XMSC\_WMQ\_MQMD\_READ\_ENABLED est définie sur true, quelles que soient les propriétés définies par l'application. Une application ne peut pas modifier les propriétés d'un message reçu, sauf si toutes les propriétés sont d'abord effacées, en fonction de la spécification JMS. Les messages reçus peuvent être réacheminés sans modification des propriétés.

**Remarque :** Si votre application reçoit un message depuis une destination avec la propriété XMSC\_WMQ\_MQMD\_READ\_ENABLED définie sur true, et qu'elle le réachemine vers une destination avec XMSC\_WMQ\_MQMD\_WRITE\_ENABLED définie sur true, toutes les valeurs de zone MSMD du message reçu sont copiées dans le message réacheminé. Table des propriétés

Tableau 878. Propriétés de l'objet Message représentant les zones MQMD		
Propriété	Description	Tapez
JMS_IBM_MQMD_REPORT	Options des messages de rapport	System.Int32
JMS_IBM_MQMD_MSGTYPE	Type de message	System.Int32
JMS_IBM_MQMD_EXPIRY	Durée de vie du message	System.Int32
JMS_IBM_MQMD_FEEDBACK	Commentaires en retour ou code anomalie	System.Int32
JMS_IBM_MQMD_ENCODING	Codage numérique des données de message	System.Int32
JMS_IBM_MQMD_CODEDCHARSETID	Identificateur de jeu de caractères des données de message	System.Int32
JMS_IBM_MQMD_FORMAT	Nom de format des données de message	System.String
JMS_IBM_MQMD_PRIORITY	Priorité de message	System.Int32
	<b>Remarque :</b> Toutes valeur affectée à JMS_IBM_MQMD_PRIORITY non comprise entre 0 et 9 enfreint la spécification JMS.	
JMS_IBM_MQMD_PERSISTENCE	Persistance des messages	System.Int32
JMS_IBM_MQMD_MSGID	Identificateur de message	Tableau d'octets
	<b>Remarque :</b> La spécification JMS indique que l'ID message doit être défini par le fournisseur JMS et qu'il doit être unique ou Null. Si vous affectez une valeur à JMS_IBM_MQMD_MSGID, celle-ci est copiée dans JMSMessageID. Elle n'est alors pas définie par le fournisseur JMS et peut ne pas être unique : cette valeur enfreint la spécification JMS.	<b>Remarque :</b> L'utilisation des propriétés de tableau d'octets sur un message enfreint la spécification JMS.
JMS_IBM_MQMD_CORRELID	Identificateur de corrélation	Tableau d'octets
	<b>Remarque :</b> Si vous affectez une valeur à JMS_IBM_MQMD_CORRELID qui commence par la chaîne "ID:", cette valeur enfreint la spécification JMS.	<b>Remarque :</b> L'utilisation des propriétés de tableau d'octets sur un message enfreint la spécification JMS.
JMS_IBM_MQMD_BACKOUTCOUNT	Nombre d'annulations	System.Int32

Tableau 878. Propriétés de l'objet Message représentant les zones MQMD (suite)

Propriété	Description	Tapez
JMS_IBM_MQMD_REPLYTOQ	Nom de la file d'attente de réponses	System.String
JMS_IBM_MQMD_REPLYTOQMGR	Nom du gestionnaire de files d'attente de réponses	System.String
JMS_IBM_MQMD_USERIDENTIFIER	ID utilisateur	System.String
JMS_IBM_MQMD_ACCOUNTINGTOKEN	Jeton de comptabilité	Tableau d'octets <b>Remarque :</b> L'utilisation des propriétés de tableau d'octets sur un message enfreint la spécification JMS.
JMS_IBM_MQMD_APPLIDENTITYDATA	Données d'application relatives à l'identité	System.String
JMS_IBM_MQMD_PUTAPPLTYPE	Type de l'application ayant placé le message en file d'attente	System.Int32
JMS_IBM_MQMD_PUTAPPLNAME	Nom de l'application ayant placé le message en file d'attente	System.String
JMS_IBM_MQMD_PUTDATE	Date à laquelle le message a été placé en file d'attente	System.String
JMS_IBM_MQMD_PUTTIME	Heure à laquelle le message a été placé en file d'attente	System.String
JMS_IBM_MQMD_APPLORIGINDATA	Données d'application relatives à l'origine	System.String
JMS_IBM_MQMD_GROUPID	Identificateur de groupe	Tableau d'octets <b>Remarque :</b> L'utilisation des propriétés de tableau d'octets sur un message enfreint la spécification JMS.
JMS_IBM_MQMD_MSGSEQNUMBER	Numéro de séquence du message local dans le groupe	System.Int32
JMS_IBM_MQMD_OFFSET	Décalage des données du message physique à partir du début du message logique	System.Int32
JMS_IBM_MQMD_MSGFLAGS	Indicateurs de message	System.Int32
JMS_IBM_MQMD_ORIGINALLENGTH	Longueur du message d'origine	System.Int32

Voir [MQMD](#) pour plus de détails.



## Exemples

Cet exemple génère un message placé dans une file d'attente ou une rubrique avec MQMD.UserIdentifier défini sur "JoeBloggs".

```
// Create a ConnectionFactory, connection, session, producer, message
// ...

// Create a destination
// ...

// Enable MQMD write
dest.setBooleanProperty(XMSC_WMQ_MQMD_WRITE_ENABLED,
    XMSC_WMQ_MQMD_WRITE_ENABLED_YES);

// Optionally, set a message context if applicable for this MD field
dest.setIntProperty(XMSC_WMQ_MQMD_MESSAGE_CONTEXT,
    XMSC_WMQ_MDCTX_SET_IDENTITY_CONTEXT);

// On the message, set property to provide custom UserId
msg.setStringProperty(JMS_IBM_MQMD_USERIDENTIFIER, "JoeBloggs");

// Send the message
// ...
```

Il est nécessaire de définir XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT avant JMS\_IBM\_MQMD\_USERIDENTIFIER. Pour plus d'informations sur l'utilisation de XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT, voir les propriétés de l'objet Message.

De même, vous pouvez extraire le contenu des zones MQMD en définissant XMSC\_WMQ\_MQMD\_READ\_ENABLED à true avant la réception d'un message, puis en utilisant les méthodes get du message telles que getStringProperty. Les propriétés reçues sont en lecture seule.

La zone de valeur prend la valeur de la zone MQMD.ApplIdentityData d'un message à partir d'une file d'attente ou d'une rubrique.

```
// Create a ConnectionFactory, connection, session, consumer
// ...

// Create a destination
// ...

// Enable MQMD read
dest.setBooleanProperty(XMSC_WMQ_MQMD_READ_ENABLED, XMSC_WMQ_MQMD_READ_ENABLED_YES);

// Receive a message
// ...

// Get required MQMD field value using a property
System.String value = rcvMsg.getStringProperty(JMS_IBM_MQMD_APPLIDENTITYDATA);
```

## Propriétés de MessageConsumer

Présentation des propriétés de l'objet MessageConsumer, avec des liens vers des informations de référence plus détaillées.

Nom de la propriété	Description
<a href="#">XMSC_IS_SUBSCRIPTION_MULTICAST</a>	Indique si les messages sont distribués au consommateur de message à l'aide de WebSphere MQ Multicast Transport. Cette propriété est en lecture seule.
<a href="#">XMSC_IS_SUBSCRIPTION_RELIABLE_MULTICAST</a>	Indique si les messages sont distribués au consommateur de message à l'aide de WebSphere MQ Multicast Transport avec une qualité de service fiable. Cette propriété est en lecture seule.

Pour plus de détails, voir [Propriétés .NET de IMessageConsumer](#).

## Propriétés de MessageProducer

Présentation des propriétés de l'objet MessageProducer, avec des liens vers des informations de référence plus détaillées.

Voir [.Propriétés NET de IMessageProducer](#) pour plus de détails.

## Propriétés de Session

Présentation des propriétés de l'objet Session, avec des liens vers des informations de référence plus détaillées.

Voir [.Propriétés NET d'ISession](#) pour plus de détails.

## Définitions de propriété

Cette section fournit une définition de chaque propriété d'objet.

Chaque définition de propriété inclut les informations suivantes :

- Le type de données de la propriété
- Les types d'objet de la propriété
- Pour une propriété de Destination, le nom qui peut être utilisé dans un identificateur URI
- Une description plus détaillée de la propriété
- Les valeurs admises de la propriété
- La valeur par défaut de la propriété

Les propriétés dont le nom commence par un des préfixes suivants sont appropriées uniquement pour le type de connexion spécifié :

### **XMSC\_RTT**

Les propriétés sont appropriées uniquement pour une connexion en temps réelle à un courtier. Les noms de propriété sont définis en tant que constantes nommées dans le fichier d'en-tête `xmsc_rtt.h`.

### **XMSC\_WMQ**

Les propriétés sont pertinentes uniquement lorsqu'une application se connecte à un gestionnaire de files d'attente IBM MQ . Les noms de propriété sont définis en tant que constantes nommées dans le fichier d'en-tête `xmsc_wmq.h`.

### **XMSC\_WPM**

Les propriétés sont appropriées uniquement lorsqu'une application se connecte à un bus d'intégration de services WebSphere. Les noms de propriété sont définis en tant que constantes nommées dans le fichier d'en-tête `xmsc_wpm.h`.

Sauf indication contraire dans leur définition, les propriétés restantes sont pertinentes pour tous les types de connexion. Les noms de propriété sont définis en tant que constantes nommées dans le fichier d'en-tête `xmsc.h`. Les propriétés dont le nom commence par le préfixe JMSX sont les propriétés de message définies par JMS, et les propriétés dont le nom commence par le préfixe JMS\_IBM sont propriétés de message définies par IBM. Pour plus d'informations sur les propriétés des messages, voir [Propriétés d'un message XMS](#).

Sauf indication contraire dans sa définition, chaque propriété est pertinente à la fois dans les domaines point à point et dans les domaines d'abonnement de publication.

Une application peut extraire et définir la valeur de toute propriété, sauf si celle-ci est désignée comme étant en lecture seule.

## **JMS\_IBM\_CHARACTER\_SET**

### **Type de données :**

System.Int32

### **Propriété de :**

Message

Identificateur (CCSID) du jeu de caractères codés, ou page de codes, dans lequel se trouvent les chaînes de données de caractères dans le corps du message lorsque le client XMS transmet le message à la destination prévue. Dans XMS, cette propriété possède une valeur numérique et des mappes sur CCSID. Toutefois, cette propriété est basée sur une propriété JMS qui contient une valeur de type chaîne et mappe vers le jeu de caractères Java qui représente ce CCSID numérique. Cette propriété remplace tout CCSID spécifié pour la destination par la propriété XMSC\_WMQ\_CCSD .

Par défaut, la propriété n'est pas définie.

Cette propriété n'est pas pertinente lorsqu'une application se connecte à un bus d'intégration de services.

## **JMS\_IBM\_ENCODING**

### **Type de données :**

System.Int32

### **Propriété de :**

Message

Mode de représentation des données numériques dans le corps du message lorsque le client XMS transmet le message à sa destination prévue. Cette propriété remplace tout codage spécifié pour la destination par la propriété XMSC\_WMQ\_ENCODING . La propriété spécifie la représentation d'entiers binaires, d'entiers décimaux condensés et de nombres à virgule flottante.

Les valeurs valides de la propriété sont identiques à celles qui peuvent être spécifiées dans la zone **Encoding** d'un descripteur de message.

Une application peut utiliser les constantes nommées suivantes pour définir la propriété :

<b>Constante nommée</b>	<b>Explication</b>
MQENC_INTEGER_NORMAL	Codage d'entier normal
MQENC_INTEGER_REVERSED	Codage d'entier inversé
MQENC_DECIMAL_NORMAL	Codage décimal condensé normal
MQENC_DECIMAL_REVERSED	Codage décimal condensé inversé
MQENC_FLOAT_IEEE_NORMAL	Codage à virgule flottante IEEE normal
MQENC_FLOAT_IEEE_REVERSED	Codage à virgule flottante IEEE inversé
MQENC_FLOAT_S390	z/OS Codage à virgule flottante à architecture
MQENC_NATIVE	Codage machine natif

Pour former une valeur pour la propriété, l'application peut ajouter trois des constantes suivantes :

- Une constante dont le nom commence par MQENC\_INTEGER, pour spécifier la représentation d'entiers binaires
- Une constante dont le nom commence par MQENC\_DECIMAL, pour spécifier la représentation d'entiers décimaux condensés
- Une constante dont le nom commence par MQENC\_FLOAT, pour spécifier la représentation de nombres à virgule flottante

L'application peut également définir la propriété à MQENC\_NATIVE, dont la valeur dépend de l'environnement.

Par défaut, la propriété n'est pas définie.

Cette propriété n'est pas pertinente lorsqu'une application se connecte à un bus d'intégration de services.

### ***JMS\_IBM\_EXCEPTIONMESSAGE***

**Type de données :**

String

**Propriété de :**

Message

Texte décrivant la raison pour laquelle le message a été envoyé à la destination d'exception. Cette propriété est en lecture seule.

Cette propriété est pertinente uniquement lorsqu'une application se connecte à un bus d'intégration de services et reçoit un message d'une destination d'exception.

### ***JMS\_IBM\_EXCEPTIONPROBLEMDESTINATION***

**Type de données :**

String

**Propriété de :**

Message

Nom de la destination du message avant que celui-ci soit envoyé à la destination d'exception.

Cette propriété est pertinente uniquement lorsqu'une application se connecte à un bus d'intégration de services et reçoit un message d'une destination d'exception.

### ***JMS\_IBM\_EXCEPTIONREASON***

**Type de données :**

System.Int32

**Propriété de :**

Message

Code anomalie indiquant la raison pour laquelle le message a été envoyé à la destination d'exception.

Cette propriété est pertinente uniquement lorsqu'une application se connecte à un bus d'intégration de services et reçoit un message d'une destination d'exception.

### ***JMS\_IBM\_EXCEPTIONTIMESTAMP***

**Type de données :**

System.Int64

**Propriété de :**

Message

Heure à laquelle le message a été envoyé à la destination d'exception.

L'heure est exprimée en millisecondes, à partir du 1er janvier 1970 00:00:00 GMT.

Cette propriété est pertinente uniquement lorsqu'une application se connecte à un bus d'intégration de services et reçoit un message d'une destination d'exception.

### ***JMS\_IBM\_FEEDBACK***

**Type de données :**

System.Int32

**Propriété de :**

Message

Code qui indique la nature d'un message de rapport.

Les valeurs valides de la propriété sont les codes retour et les codes raison qui peuvent être spécifiés dans la zone **Feedback** d'un descripteur de message.

Par défaut, la propriété n'est pas définie.

### ***JMS\_IBM\_FORMAT***

**Type de données :**

String

**Propriété de :**

Message

Nature des données d'application dans le message.

Les valeurs valides de la propriété sont identiques à celles qui peuvent être spécifiées dans la zone **Format** d'un descripteur de message.

Par défaut, la propriété n'est pas définie.

Cette propriété n'est pas pertinente lorsqu'une application se connecte à un bus d'intégration de services.

### ***JMS\_IBM\_LAST\_MSG\_IN\_GROUP***

**Type de données :**

System.Boolean

**Propriété de :**

Message

Indique si le message est le dernier d'un groupe de messages.

Définissez la propriété à true si le message est le dernier d'un groupe de messages. Dans le cas contraire, définissez la propriété à false ou ne la définissez pas. Par défaut, la propriété n'est pas définie.

La valeur true correspond à l'indicateur de statut MQMF\_LAST\_MSG\_IN\_GROUP, qui peut être spécifié dans la zone **MsgFlags** d'un descripteur de message.

Cette propriété est ignorée dans le domaine de publication / abonnement et n'est pas pertinente lorsqu'une application se connecte à un bus d'intégration de services.

### ***JMS\_IBM\_MSGTYPE***

**Type de données :**

System.Int32

**Propriété de :**

Message

Type du message.

Les valeurs admises de la propriété sont les suivantes :

<b>Valeur admise</b>	<b>Explication</b>
MQMT_DATAGRAM	Le message ne requiert pas de réponse.
MQMT_REQUEST	Le message requiert une réponse.
MQMT_REPLY	Le message est un message de réponse.
MQMT_REPORT	Le message est un message de rapport.

Ces valeurs correspondent aux types de message qui peuvent être spécifiés dans la zone **MsgType** d'un descripteur de message.

Par défaut, la propriété n'est pas définie.

Cette propriété n'est pas pertinente lorsqu'une application se connecte à un bus d'intégration de services.

## ***JMS\_IBM\_PUTAPPLTYPE***

**Type de données :**

System.Int32

**Propriété de :**

Message

Type de l'application ayant envoyé le message.

Les valeurs admises sont les types d'application qui peuvent être spécifiés dans la zone **PutApp1Type** d'un descripteur de message.

Par défaut, la propriété n'est pas définie.

Cette propriété n'est pas pertinente lorsqu'une application se connecte à un bus d'intégration de services.

## ***JMS\_IBM\_PUTDATE***

**Type de données :**

String

**Propriété de :**

Message

Date d'envoi du message.

Les valeurs valides de la propriété sont identiques à celles qui peuvent être spécifiées dans la zone **PutDate** d'un descripteur de message.

Par défaut, la propriété n'est pas définie.

Cette propriété n'est pas pertinente lorsqu'une application se connecte à un bus d'intégration de services.

## ***JMS\_IBM\_PUTTIME***

**Type de données :**

String

**Propriété de :**

Message

Heure d'envoi du message.

Les valeurs valides de la propriété sont identiques à celles qui peuvent être spécifiées dans la zone **PutTime** d'un descripteur de message.

Par défaut, la propriété n'est pas définie.

Cette propriété n'est pas pertinente lorsqu'une application se connecte à un bus d'intégration de services.

## ***JMS\_IBM\_REPORT\_COA***

**Type de données :**

System.Int32

**Propriété de :**

Message

Demande de messages de rapport "confirmation à l'arrivée", avec indication de la quantité de données d'application du message d'origine devant être incluse dans un message de rapport.

Les valeurs admises de la propriété sont les suivantes :

**Valeur admise**

MQRO\_COA

**Explication**

Demande de messages de rapport "confirmation à l'arrivée", sans données d'application du message d'origine incluses dans un message de rapport.

**Valeur admise**

MQRO\_COA\_WITH\_DATA

**Explication**

Demande de messages de rapport "confirmation à l'arrivée", avec les 100 premiers octets de données d'application du message d'origine incluses dans un message de rapport.

MQRO\_COA\_WITH\_FULL\_DATA

Demande de messages de rapport "confirmation à l'arrivée", avec toutes les données d'application du message d'origine incluses dans un message de rapport.

Ces valeurs correspondent aux options de rapport qui peuvent être spécifiées dans la zone **Report** d'un descripteur de message. Pour plus d'informations sur ces options, voir [Rapport \(MQLONG\)](#).

Par défaut, la propriété n'est pas définie.

**JMS\_IBM\_REPORT\_COD****Type de données :**

System.Int32

**Propriété de :**

Message

Demande de messages de rapport "confirmation à la livraison", avec indication de la quantité de données d'application du message d'origine devant être incluse dans un message de rapport.

Les valeurs admises de la propriété sont les suivantes :

**Valeur admise**

MQRO\_COD

**Explication**

Demande de messages de rapport "confirmation à la livraison", sans données d'application du message d'origine incluses dans un message de rapport.

MQRO\_COD\_WITH\_DATA

Demande de messages de rapport "confirmation à la livraison", avec les 100 premiers octets de données d'application du message d'origine incluses dans un message de rapport.

MQRO\_COD\_WITH\_FULL\_DATA

Demande de messages de rapport "confirmation à la livraison", avec toutes les données d'application du message d'origine incluses dans un message de rapport.

Ces valeurs correspondent aux options de rapport qui peuvent être spécifiées dans la zone **Report** d'un descripteur de message.

Par défaut, la propriété n'est pas définie.

**JMS\_IBM\_REPORT\_DISCARD\_MSG****Type de données :**

System.Int32

**Propriété de :**

Message

Demande de destruction du message au cas où celui-ci ne peut pas être livré à la destination prévue.

Définissez la propriété à MQRO\_DISCARD\_MSG pour demander la destruction du message au cas où celui-ci ne peut être livré à la destination prévue. Si vous souhaitez que le message soit placé dans une file d'attente de messages non livrés ou envoyé à une destination d'exception, ne définissez pas cette propriété. Par défaut, la propriété n'est pas définie.

La valeur MQRO\_DISCARD\_MSG correspond à une option de rapport qui peut être spécifiée dans la zone **Report** d'un descripteur de message.

## ***JMS\_IBM\_REPORT\_EXCEPTION***

### **Type de données :**

System.Int32

### **Propriété de :**

Message

Demande de messages de rapport d'exception, avec indication de la quantité de données d'application du message d'origine devant être incluse dans un message de rapport.

Les valeurs admises de la propriété sont les suivantes :

<b>Valeur admise</b>	<b>Explication</b>
MQRO_EXCEPTION	Demande de messages de rapport d'exception, sans données d'application du message d'origine incluses dans un message de rapport.
MQRO_EXCEPTION_WITH_DATA	Demande de messages de rapport d'exception, avec les 100 premiers octets de données d'application du message d'origine incluses dans un message de rapport.
MQRO_EXCEPTION_WITH_FULL_DATA	Demande de messages de rapport d'exception, avec toutes les données d'application du message d'origine incluses dans un message de rapport.

Ces valeurs correspondent aux options de rapport qui peuvent être spécifiées dans la zone **Report** d'un descripteur de message.

Par défaut, la propriété n'est pas définie.

## ***JMS\_IBM\_REPORT\_EXPIRATION***

### **Type de données :**

System.Int32

### **Propriété de :**

Message

Demande de messages de rapport d'expiration, avec indication de la quantité de données d'application du message d'origine devant être incluse dans un message de rapport.

Les valeurs admises de la propriété sont les suivantes :

<b>Valeur admise</b>	<b>Explication</b>
MQRO_EXPIRATION	Demande de messages de rapport d'expiration, sans données d'application du message d'origine incluses dans un message de rapport.
MQRO_EXPIRATION_WITH_DATA	Demande de messages de rapport d'expiration, avec les 100 premiers octets de données d'application du message d'origine incluses dans un message de rapport.
MQRO_EXPIRATION_WITH_FULL_DATA	Demande de messages de rapport d'expiration, avec toutes les données d'application du message d'origine incluses dans un message de rapport.

Ces valeurs correspondent aux options de rapport qui peuvent être spécifiées dans la zone **Report** d'un descripteur de message.



Par défaut, la propriété n'est pas définie.

### ***JMS\_IBM\_REPORT\_NAN***

**Type de données :**

System.Int32

**Propriété de :**

Message

Demande de messages de rapport de notification d'action négative.

Définissez la propriété à MQRO\_NAN pour demander des messages de rapport de notification d'action négative. Si vous ne voulez pas de des messages de rapport de notification d'action négative, ne définissez pas cette propriété. Par défaut, la propriété n'est pas définie.

La valeur MQRO\_NAN correspond à une option de rapport qui peut être spécifiée dans la zone **Report** d'un descripteur de message.

### ***JMS\_IBM\_REPORT\_PAN***

**Type de données :**

System.Int32

**Propriété de :**

Message

Demande de messages de rapport de notification d'action positive.

Définissez la propriété à MQRO\_PAN pour demander des messages de rapport de notification d'action positive. Si vous ne voulez pas de des messages de rapport de notification d'action positive, ne définissez pas cette propriété. Par défaut, la propriété n'est pas définie.

La valeur MQRO\_PAN correspond à une option de rapport qui peut être spécifiée dans la zone **Report** d'un descripteur de message.

### ***JMS\_IBM\_REPORT\_PASS\_CORREL\_ID***

**Type de données :**

System.Int32

**Propriété de :**

Message

Demande que l'identificateur de corrélation de tout message de rapport ou de réponse soit le même que celui du message d'origine.

Les valeurs admises de la propriété sont les suivantes :

<b>Valeur admise</b>	<b>Explication</b>
MQRO_PASS_CORREL_ID	Demande selon laquelle l'identificateur de corrélation de tout message de rapport ou de réponse doit être le même que celui du message d'origine.
MQRO_COPY_MSG_ID_TO_CORREL_ID	Demande que l'identificateur de corrélation de tout message de rapport ou de réponse soit le même que l'identificateur de message du message d'origine.

Ces valeurs correspondent aux options de rapport qui peuvent être spécifiées dans la zone **Report** d'un descripteur de message.

La valeur par défaut de la propriété est MQRO\_COPY\_MSG\_ID\_TO\_CORREL\_ID.

## **JMS\_IBM\_REPORT\_PASS\_MSG\_ID**

**Type de données :**

System.Int32

**Propriété de :**

Message

Demande que l'identificateur de message de tout message de rapport ou de réponse soit le même que celui du message d'origine.

Les valeurs admises de la propriété sont les suivantes :

<b>Valeur admise</b>	<b>Explication</b>
MQRO_PASS_MSG_ID	Demande selon laquelle l'identificateur de message de tout message de rapport ou de réponse doit être le même que celui du message d'origine.
MQRO_NEW_MSG_ID	Demande qu'un nouvel identificateur de message soit généré pour chaque message de rapport ou de réponse.

Ces valeurs correspondent aux options de rapport qui peuvent être spécifiées dans la zone Rapport d'un descripteur de message.

La valeur par défaut de la propriété est MQRO\_NEW\_MSG\_ID.

## **JMS\_IBM\_RETAIN**

**Type de données :**

System.Int32

**Propriété de :**

Message

La définition de cette propriété indique que le gestionnaire de file d'attente traite un message en tant que publication conservée. Lorsqu'un abonné reçoit des messages provenant de rubriques, il peut recevoir des messages supplémentaires immédiatement après l'abonnement, en plus de ceux reçus dans les versions précédentes. Ces messages sont les publications conservées (facultatives) pour les rubriques concernées par l'abonnement. Pour chaque rubrique correspondant à l'abonnement, les publications conservées sont disponibles pour livraison au consommateur de message abonné.

RETAIN\_PUBLICATION est la seule valeur admise pour cette propriété. Par défaut, cette propriété n'est pas définie.

**Remarque :** Cette propriété est pertinente uniquement dans le domaine publication/abonnement.

## **JMS\_IBM\_SYSTEM\_MESSAGEID**

**Type de données :**

String

**Propriété de :**

Message

Identificateur unique du message dans le bus d'intégration de services. Cette propriété est en lecture seule.

Cette propriété est pertinente uniquement lorsqu'une application se connecte à un bus d'intégration de services.

## **JMSX\_APPID**

**Type de données :**

String

**Propriété de :**

Message

Nom de l'application ayant envoyé le message.

Il s'agit de la propriété JMS définie avec le nom JMS JMSXAppID. Pour plus d'informations sur cette propriété, voir *Java Message Service Specification, Version 1.1*.

Par défaut, la propriété n'est pas définie.

Cette propriété n'est pas valide pour une connexion en temps réel à un courtier.

**JMSX\_DELIVERY\_COUNT****Type de données :**

System.Int32

**Propriété de :**

Message

Nombre de tentatives de livraison du message.

Il s'agit de la propriété JMS définie avec le nom JMS JMSXDeliveryCount. Pour plus d'informations sur la propriété, voir *Java Message Service Specification, Version 1.1*.

Par défaut, la propriété n'est pas définie.

Cette propriété n'est pas valide pour une connexion en temps réel à un courtier.

**JMSX\_GROUPID****Type de données :**

String

**Propriété de :**

Message

Identificateur du groupe de messages auquel le message appartient.

Il s'agit de la propriété JMS définie avec le nom JMS JMSXGroupID. Pour plus d'informations sur la propriété, voir *Java Message Service Specification, Version 1.1*.

Par défaut, la propriété n'est pas définie.

Cette propriété n'est pas valide pour une connexion en temps réel à un courtier.

**JMSX\_GROUPSEQ****Type de données :**

System.Int32

**Propriété de :**

Message

Numéro de séquence du message dans un groupe de messages.

Il s'agit de la propriété JMS définie avec le nom JMS JMSXGroupSeq. Pour plus d'informations sur la propriété, voir *Java Message Service Specification, Version 1.1*.

Par défaut, la propriété n'est pas définie.

Cette propriété n'est pas valide pour une connexion en temps réel à un courtier.

**JMSX\_USERID****Type de données :**

String

**Propriété de :**

Message

ID utilisateur associé à l'application ayant envoyé le message.

Il s'agit de la propriété JMS définie avec le nom JMS JMSXUserID. Pour plus d'informations sur la propriété, voir *Java Message Service Specification, Version 1.1*.

Par défaut, la propriété n'est pas définie.

Cette propriété n'est pas valide pour une connexion en temps réel à un courtier.

## ***XMSC\_ASYNC\_EXCEPTIONS***

### **Type de données :**

System.Int32

### **Propriété de :**

ConnectionFactory

### **Objets applicables:**

Nom long de l'outil d'administration JMS: ASYNCEXCEPTION

Nom abrégé de l'outil d'administration JMS: AEX

Cette propriété détermine si XMS informe un programme d'écoute des exceptions uniquement lorsqu'une connexion est interrompue, ou également lorsqu'une exception est émise de façon asynchrone sur un appel d'API XMS. Cette propriété s'applique à toutes les connexions créées à partir de la fabrique de connexions et pour lesquelles un programme d'écoute des exceptions est enregistré.

Les valeurs valides de cette propriété sont les suivantes :

### ***XMSC\_ASYNC\_EXCEPTIONS\_ALL***

Toutes les exceptions détectées de façon asynchrone, en dehors de la portée d'un appel d'API synchrone, et toutes les exceptions de connexion interrompue sont envoyées au programme d'écoute des exceptions.

### ***XMSC\_ASYNC\_EXCEPTIONS\_CONNECTIONBROKEN***

Seules les exceptions signalant une connexion interrompue sont envoyées au programme d'écoute des exceptions. Toutes les autres exceptions se produisant au cours d'un traitement asynchrone ne sont pas rapportées au programme d'écoute des exceptions ; l'application n'est donc pas informée de ces exceptions.

Par défaut, cette propriété est définie sur XMSC\_ASYNC\_EXCEPTIONS\_ALL.

## ***XMSC\_CLIENT\_ID***

### **Type de données :**

String

### **Propriété de :**

ConnectionFactory

### **Objets applicables:**

Nom long de l'outil d'administration JMS: CLIENTID

Nom abrégé de l'outil d'administration JMS: CID

Identificateur de client pour une connexion.

Un identificateur de client est utilisé uniquement pour prendre en charge des abonnements durables dans le domaine publication/abonnement ; il est ignoré dans le domaine point-à-point. Pour plus d'informations sur la définition des identificateurs client, voir [ConnectionFactoryes](#) et [Connection objects](#).

Cette propriété n'est pas pertinente pour une connexion en temps réel à un courtier.

## ***XMSC\_CONNECTION\_TYPE***

### **Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory

Type de serveur de messagerie auquel une application se connecte.

Les valeurs admises de la propriété sont les suivantes :

<b>Valeur admise</b>	<b>Explication</b>
XMSC_CT_RTT	Connexion en temps réel à un courtier.
XMSC_CT_WMQ	Connexion à un gestionnaire de files d'attente IBM MQ .
XMSC_CT_WPM	Une connexion à un WebSphere Application Server service integration bus.

Par défaut, la propriété n'est pas définie.

***XMSC\_DELIVERY\_MODE*****Type de données :**

System.Int32

**Propriété de :**

Destination

**Nom utilisé dans un URI :**

    persistence (pour une destination IBM MQ )

    deliveryMode (pour une destination de fournisseur de messagerie par défaut WebSphere)

**Objets applicables:**

    Nom long de l'outil d'administration JMS: PERSISTENCE

    Nom abrégé de l'outil d'administration JMS: PER

Mode de livraison des messages envoyés à la destination.

Les valeurs admises de la propriété sont les suivantes :

<b>Valeur admise</b>	<b>Explication</b>
XMSC_DELIVERY_NOT_PERSISTENT	Un message envoyé à la destination est non permanent. Le mode de livraison par défaut de l'expéditeur de message, ou tout mode de livraison spécifié sur l'appel Send, est ignoré. Si la destination est une file d'attente IBM MQ , la valeur de l'attribut de file d'attente <i>DefPersistence</i> est également ignorée.
XMSC_DELIVERY_PERSISTENT	Un message envoyé à la destination est permanent. Le mode de livraison par défaut de l'expéditeur de message, ou tout mode de livraison spécifié sur l'appel Send, est ignoré. Si la destination est une file d'attente IBM MQ , la valeur de l'attribut de file d'attente <i>DefPersistence</i> est également ignorée.
XMSC_DELIVERY_AS_APP	Un message envoyé à la destination dont le mode de livraison est spécifié sur l'appel Send. Si l'appel Send ne spécifie pas de mode de livraison, le mode de livraison par défaut de l'expéditeur de message est utilisé. Si la destination est une file d'attente IBM MQ , la valeur de l'attribut de file d'attente <i>DefPersistence</i> est ignorée.

**Valeur admise**

XMSC\_DELIVERY\_AS\_DEST

**Explication**

Si la destination est une file d'attente IBM MQ , un message inséré dans la file d'attente a le mode de livraison spécifié par la valeur de l'attribut de file d'attente *DefPersistence*. Le mode de livraison par défaut de l'expéditeur de message, ou tout mode de livraison spécifié sur l'appel Send, est ignoré.

Si la destination n'est pas une file d'attente IBM MQ , la signification est la même que celle de XMSC\_DELIVERY\_AS\_APP.

La valeur par défaut est XMSC\_DELIVERY\_AS\_APP.

***XMSC\_IC\_PROVIDER\_URL*****Type de données :**

String

**Propriété de :**

InitialContext

Utilisé pour localiser le répertoire de désignation JNDI de sorte que le service de dénomination COS n'ait pas besoin d'être sur le même serveur que le service Web.

***XMSC\_IC\_SECURITY\_AUTHENTICATION*****Type de données :**

String

**Propriété de :**

InitialContext

Basé sur l'interface de contexte Java SECURITY\_AUTHENTICATION. Cette propriété est applicable uniquement au contexte d'affectation de classe de service.

***XMSC\_IC\_SECURITY\_CREDENTIALS*****Type de données :**

String

**Propriété de :**

InitialContext

Basé sur l'interface de contexte Java SECURITY\_CREDENTIALS. Cette propriété est applicable uniquement au contexte d'affectation de classe de service.

***XMSC\_IC\_SECURITY\_PRINCIPAL*****Type de données :**

String

**Propriété de :**

InitialContext

Basé sur l'interface de contexte Java SECURITY\_PRINCIPAL. Cette propriété est applicable uniquement au contexte d'affectation de classe de service.

***XMSC\_IC\_SECURITY\_PROTOCOL*****Type de données :**

String

**Propriété de :**

InitialContext

Basé sur l'interface de contexte Java SECURITY\_PROTOCOL Cette propriété est applicable uniquement au contexte d'affectation de classe de service.

***XMSC\_IC\_URL*****Type de données :**

String

**Propriété de :**

InitialContext

Pour les contextes LDAP et FileSystem, adresse du référentiel contenant les objets gérés.

Pour les contextes LDAP et FileSystem, adresse du référentiel contenant les objets gérés.

***XMSC\_IS\_SUBSCRIPTION\_MULTICAST*****Type de données :**

System.Boolean

**Propriété de :**

MessageConsumer

Indique si les messages sont distribués au consommateur de message à l'aide de WebSphere MQ Multicast Transport. Cette propriété est en lecture seule.

La valeur de la propriété est true si les messages sont distribués au consommateur de message via WebSphere MQ Multicast Transport. Dans le cas contraire, la valeur est false.

Cette propriété est pertinente uniquement pour une connexion en temps réel à un courtier.

***XMSC\_IS\_SUBSCRIPTION\_RELIABLE\_MULTICAST*****Type de données :**

System.Boolean

**Propriété de :**

MessageConsumer

Indique si les messages sont distribués au consommateur de message à l'aide de WebSphere MQ Multicast Transport avec une qualité de service fiable. Cette propriété est en lecture seule.

La valeur de la propriété est true si les messages sont distribués au consommateur de message via WebSphere MQ Multicast Transport avec une qualité de service fiable. Dans le cas contraire, la valeur est false.

Cette propriété est pertinente uniquement pour une connexion en temps réel à un courtier.

***XMSC\_JMS\_MAJOR\_VERSION*****Type de données :**

System.Int32

**Propriété de :**

ConnectionMetaData

Numéro de version principale de la spécification JMS sur laquelle repose XMS . Cette propriété est en lecture seule.

***XMSC\_JMS\_MINOR\_VERSION*****Type de données :**

System.Int32

**Propriété de :**

ConnectionMetaData

Numéro de version secondaire de la spécification JMS sur laquelle repose XMS . Cette propriété est en lecture seule.

***XMSC\_JMS\_VERSION*****Type de données :**

String

**Propriété de :**

ConnectionMetaData

Identificateur de version de la spécification JMS sur laquelle est basée XMS . Cette propriété est en lecture seule.

***XMSC\_MAJOR\_VERSION*****Type de données :**

System.Int32

**Propriété de :**

ConnectionMetaData

Numéro de version du client XMS . Cette propriété est en lecture seule.

***XMSC\_MINOR\_VERSION*****Type de données :**

System.Int32

**Propriété de :**

ConnectionMetaData

Numéro d'édition du client XMS . Cette propriété est en lecture seule.

***XMSC\_PASSWORD*****Type de données :**


Tableau d'octets


**Propriété de :**

ConnectionFactory

Mot de passe pouvant être utilisé pour authentifier l'application lorsqu'elle tente de se connecter à un serveur de messagerie. Le mot de passe est utilisé avec la propriété XMSC\_USERID .

Par défaut, la propriété n'est pas définie.

 Si vous vous connectez à IBM MQ sur Multiplateformes et que vous définissez la propriété XMSC\_USERID de la fabrique de connexions, elle doit correspondre au **userid** de l'utilisateur connecté. Si vous ne définissez pas ces propriétés, le gestionnaire de files d'attente utilise par défaut le **userid** de l'utilisateur connecté. Si vous avez besoin d'authentifications de niveau connexion pour des utilisateurs individuels, vous pouvez écrire un exit d'authentification de client, configuré dans IBM MQ. Pour plus d'informations sur la création d'un exit d'authentification de client, voir Planification de l'authentification pour une application client.

 Pour authentifier l'utilisateur lors de la connexion à IBM MQ for z/OS, vous devez utiliser un exit de sécurité.

***XMSC\_PRIORITY*****Type de données :**

System.Int32



**Propriété de :**

Destination

**Nom utilisé dans un URI :**

priority

Priorité des messages envoyés à la destination.

Les valeurs admises de la propriété sont les suivantes :

<b>Valeur admise</b>	<b>Explication</b>
Un entier, compris entre 0, priorité la plus faible, et 9, priorité la plus élevée	Un message envoyé à la destination a la priorité spécifiée. La priorité par défaut de l'expéditeur de message, ou toute priorité spécifiée sur l'appel Send, est ignorée. Si la destination est une file d'attente IBM MQ , la valeur de l'attribut de file d'attente <b>DefPriority</b> est également ignorée.
XMSC_PRIORITY_AS_APP	Un message envoyé à la destination a la priorité spécifiée sur l'appel Send. Si l'appel Send ne spécifie aucune priorité, la priorité par défaut de l'expéditeur de message est utilisée. Si la destination est une file d'attente IBM MQ , la valeur de l'attribut de file d'attente <b>DefPriority</b> est ignorée.
XMSC_PRIORITY_AS_DEST	Si la destination est une file d'attente IBM MQ , un message inséré dans la file d'attente a la priorité spécifiée par la valeur de l'attribut de file d'attente <b>DefPriority</b> . La priorité par défaut de l'expéditeur de message, ou toute priorité spécifiée sur l'appel Send, est ignorée.  Si la destination n'est pas une file d'attente IBM MQ , la signification est la même que celle de XMSC_PRIORITY_AS_APP.

La valeur par défaut est XMSC\_PRIORITY\_AS\_APP.

WebSphere MQ Real-Time Transport et WebSphere MQ Multicast Transport n'effectuent aucune action basée sur la priorité d'un message.

***XMSC\_PROVIDER\_NAME*****Type de données :**

String

**Propriété de :**

ConnectionMetaData

Fournisseur du client XMS . Cette propriété est en lecture seule.

***XMSC\_RTT\_BROKER\_PING\_INTERVAL*****Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory

Intervalle de temps, en millisecondes, après lequel XMS .NET vérifie la connexion à un serveur de messagerie en temps réel pour détecter une activité. Si aucune activité n'est détectée, le client initie une commande ping ; la connexion est fermée si aucune réponse à la commande ping n'est détectée.

La valeur par défaut de la propriété est 30000.

***XMSC\_RTT\_CONNECTION\_PROTOCOL*****Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory

Protocole de communication utilisé pour une connexion en temps réel à un courtier.

La valeur de la propriété doit être `XMSC_RTT_CP_TCP`, ce qui indique une connexion en temps réel à un courtier via TCP/IP. La valeur par défaut est `XMSC_RTT_CP_TCP`.

***XMSC\_RTT\_HOST\_NAME*****Type de données :**

String

**Propriété de :**

ConnectionFactory

Nom d'hôte ou adresse IP du système sur lequel un courtier s'exécute.

Cette propriété est utilisée avec la propriété `XMSC_RTT_PORT` pour identifier le courtier.

Par défaut, la propriété n'est pas définie.

***XMSC\_RTT\_LOCAL\_ADDRESS*****Type de données :**

String

**Propriété de :**

ConnectionFactory

Nom d'hôte ou adresse IP de l'interface de réseau local à utiliser pour une connexion en temps réel à un courtier.

Cette propriété est utile uniquement si le système sur lequel l'application est exécutée dispose d'au moins deux interfaces réseau et que vous devez pouvoir spécifier l'interface qui doit être utilisée pour une connexion en temps réel. Si le système dispose d'une seule interface réseau, seule cette interface peut être utilisée. Si le système dispose d'au moins deux interfaces réseau et que la propriété n'est pas définie, l'interface est sélectionnée au hasard.

Par défaut, la propriété n'est pas définie.

***XMSC\_RTT\_MULTICAST*****Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory et Destination

**Nom utilisé dans un URI :**

multicast

Paramètre de multidiffusion pour une fabrique de connexions ou une destination. Seule une destination représentant une rubrique peut avoir cette propriété.

Une application utilise cette propriété pour activer la multidiffusion en association avec une connexion en temps réel à un courtier et, si la multidiffusion est activée, pour spécifier la manière précise dont la multidiffusion est utilisée pour distribuer les messages depuis un courtier vers un consommateur de messages. La propriété n'a pas d'effet sur la manière dont un expéditeur de message envoie des messages au courtier.

Les valeurs admises de la propriété sont les suivantes :

**Valeur admise**

XMSC\_RTT\_MULTICAST\_DISABLED

XMSC\_RTT\_MULTICAST\_ASCF

XMSC\_RTT\_MULTICAST\_ENABLED

XMSC\_RTT\_MULTICAST\_RELIABLE

XMSC\_RTT\_MULTICAST\_NOT\_RELIABLE

**Explication**

Les messages ne sont pas distribués à un consommateur de message à l'aide de WebSphere MQ Multicast Transport. Il s'agit de la valeur par défaut pour un objet ConnectionFactory.

Les messages sont distribués à un consommateur de message en fonction du paramètre de multidiffusion défini pour la fabrique de connexions associée au consommateur de message. Le paramètre de multidiffusion défini pour la fabrique de connexions est déterminé au moment de la création de la connexion. Cette valeur est admise uniquement pour un objet Destination, dont elle est la valeur par défaut.

Si la rubrique est configurée pour la multidiffusion dans le courtier, les messages sont distribués à un consommateur de message à l'aide de WebSphere MQ Multicast Transport. Une qualité de service fiable est utilisée si la rubrique est configurée pour une multidiffusion fiable.

Si la rubrique est configurée pour la multidiffusion fiable dans le courtier, les messages sont distribués à un consommateur de message à l'aide de WebSphere MQ Multicast Transport avec une qualité de service fiable. Si la rubrique n'est pas configurée pour la multidiffusion fiable, vous ne pouvez pas créer de consommateur de message pour la rubrique.

Si la rubrique est configurée pour la multidiffusion dans le courtier, les messages sont distribués à un consommateur de message à l'aide de WebSphere MQ Multicast Transport. Une qualité de service fiable n'est pas utilisée, même si la rubrique est configurée pour une multidiffusion fiable.

***XMSC\_RTT\_PORT*****Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory

Numéro du port sur lequel un courtier écoute les demandes entrantes. Sur le courtier, vous devez configurer un noeud de traitement de message Real-timeInput ou Real-timeOptimizedFlow pour écouter sur ce port.

Cette propriété est utilisée avec la propriété [XMSC\\_RTT\\_HOST\\_NAME](#) pour identifier le courtier.

La valeur par défaut de la propriété est XMSC\_RTT\_DEFAULT\_PORT, ou 1506.

## ***XMSC\_TIME\_TO\_LIVE***

**Type de données :**

System.Int32

**Propriété de :**

Destination

**Nom utilisé dans un URI :**

expiration (pour une destination IBM MQ )

timeToLive (pour une destination fournisseur de messagerie par défaut WebSphere)

Durée de vie des messages envoyés à la destination.

Les valeurs admises de la propriété sont les suivantes :

<b>Valeur admise</b>	<b>Explication</b>
0	Un message envoyé à la destination n'expire jamais.
Entier positif	Un message envoyé à la destination a la durée de vie spécifiée (en millisecondes). La durée de vie par défaut de l'expéditeur de message ou toute durée de vie spécifiée sur l'appel Send est ignorée.
XMSC_TIME_TO_LIVE_AS_APP	Un message envoyé à la destination a la durée de vie spécifiée sur l'appel Send. Si l'appel Send ne spécifie pas de durée de vie, la valeur par défaut de l'expéditeur de message est utilisée.

La valeur par défaut est XMSC\_TIME\_TO\_LIVE\_AS\_APP.

## ***XMSC\_USERID***

**Type de données :**


String


**Propriété de :**

ConnectionFactory

ID utilisateur pouvant être utilisé pour authentifier l'application lorsqu'elle tente de se connecter à un serveur de messagerie. L'identificateur utilisateur est utilisé avec la propriété [XMSC\\_PASSWORD](#).

Par défaut, la propriété n'est pas définie.

 Si vous vous connectez à IBM MQ for Multiplatformset que vous définissez la propriété XMSC\_USERID de la fabrique de connexions, elle doit correspondre au **userid** de l'utilisateur connecté. Si vous ne définissez pas ces propriétés, le gestionnaire de files d'attente utilise par défaut le **userid** de l'utilisateur connecté. Si vous avez besoin d'authentifications de niveau connexion pour des utilisateurs individuels, vous pouvez écrire un exit d'authentification de client, configuré dans IBM MQ. Pour plus d'informations sur la création d'un exit d'authentification de client, voir [Planification de l'authentification pour une application client](#).

 Pour authentifier l'utilisateur lors de la connexion à IBM MQ for z/OS, vous devez utiliser un exit de sécurité.

## ***XMSC\_VERSION***

**Type de données :**

String

**Propriété de :**

ConnectionMetaData

Identificateur de version de l'interface de ligne de commandeXMSent. Cette propriété est en lecture seule.

## **V 9.3.0 XMSC\_WMQ\_BALANCING\_APPLICATION\_TYPE**

### **Type de données :**

System.Int32

### **Propriété de :**

ConnectionFactory

Les valeurs admises de la propriété sont les suivantes :

#### **Valeur admise**

XMSC\_WMQ\_BALANCING\_APPLICATION\_TYPE\_SIMPLE

XMSC\_WMQ\_BALANCING\_APPLICATION\_TYPE\_REQUEST\_REPLY

#### **Explication**

Equilibrage simple ; aucune règle spécifique n'est appliquée en plus de celles décrites dans [Influencer le rééquilibrage des applications dans des clusters uniformes](#). Il s'agit de la valeur par défaut.

Equilibrage de demande-réponse ; après chaque appel MQPUT, un appel MQGET correspondant est attendu pour un message de réponse. L'équilibrage est différé jusqu'à ce qu'un tel message soit reçu ou que le message de demande EXPIRATION ait été dépassé

De plus, cette propriété peut être définie dans le fichier `client.ini`. L'ordre de préférence est le suivant:

1. Propriétés définies dans l'application
2. Mise en correspondance de la [strophe d'application](#) nommée dans le fichier `mqclient.ini`.
3. [Strophe Application defaults](#) dans le fichier `mqclient.ini`.

## **V 9.3.0 XMSC\_WMQ\_BALANCING\_OPTIONS**

### **Type de données :**

System.Int32

### **Propriété de :**

ConnectionFactory

Les valeurs admises de la propriété sont les suivantes :

#### **Valeur admise**

XMSC\_WMQ\_BALANCING\_OPTIONS\_NONE

XMSC\_WMQ\_BALANCING\_OPTIONS\_IGNORE\_TRANSACTION

#### **Valeur correspondante**

Aucune option n'est définie. Il s'agit de la valeur par défaut

La définition de cette option permet de rééquilibrer les applications même si elles se trouvent au milieu d'une transaction.

De plus, cette propriété peut être définie dans le fichier `client.ini`. L'ordre de préférence est le suivant:

1. Propriétés définies dans l'application
2. Mise en correspondance de la [strophe d'application](#) nommée dans le fichier `mqclient.ini`.

3. Strophe Application defaults dans le fichier `mqclient.ini`.

## **V 9.3.0** ***XMSC\_WMQ\_BALANCING\_TIMEOUT***

**Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory

Les valeurs admises de la propriété sont les suivantes :

**Valeur admise**

XMSC\_WMQ\_BALANCING\_TIMEOUT\_IMMEDIATE

XMSC\_WMQ\_BALANCING\_TIMEOUT\_AS\_DEFAULT

XMSC\_WMQ\_BALANCING\_TIMEOUT\_NEVER

**Explication**

Dépassement immédiat du délai d'attente

Valeur de délai d'attente par défaut définie. Il s'agit de la valeur par défaut

Aucun délai d'attente n'est dépassé

**Remarque :** Vous devez indiquer une seule valeur parmi les valeurs définies ou une valeur comprise entre 0 et 999999999 secondes.

De plus, cette propriété peut être définie dans le fichier `client.ini`. L'ordre de préférence est le suivant:

1. Propriétés définies dans l'application
2. Mise en correspondance de la strophe d'application nommée dans le fichier `mqclient.ini`.
3. Strophe Application defaults dans le fichier `mqclient.ini`.

## ***XMSC\_WMQ\_BROKER\_CONTROLQ***

**Type de données :**

String

**Propriété de :**

ConnectionFactory

Nom de la file d'attente de contrôle utilisée par un courtier.

La valeur par défaut de la propriété est `SYSTEM.BROKER.CONTROL.QUEUE`.

Cette propriété est pertinente uniquement dans le domaine publication/abonnement.

## ***XMSC\_WMQ\_BROKER\_PUBQ***

**Type de données :**

String

**Propriété de :**

ConnectionFactory

Nom de la file d'attente contrôlée par un courtier lorsque des applications envoient les messages qu'elles publient.

La valeur par défaut de la propriété est `SYSTEM.BROKER.DEFAULT.STREAM`.

Cette propriété est pertinente uniquement dans le domaine publication/abonnement.

## ***XMSC\_WMQ\_BROKER\_QMGR***

**Type de données :**

String

**Propriété de :**

ConnectionFactory

Nom du gestionnaire de files d'attente auquel un courtier est connecté.

Par défaut, la propriété n'est pas définie.

Cette propriété est pertinente uniquement dans le domaine publication/abonnement.

***XMSC\_WMQ\_BROKER\_SUBQ*****Type de données :**

String

**Propriété de :**

ConnectionFactory

Nom de la file d'attente de souscription pour un consommateur de message non durable.

Le nom de la file d'attente de souscription doit commencer par les caractères suivants :

SYSTEM.JMS.ND.

Si vous voulez que tous les consommateurs de message non durables partagent une file d'attente de souscription, spécifiez le nom complet de la file d'attente partagée. Une file d'attente du nom spécifié doit exister avant qu'une application puisse créer un consommateur de message non durable.

Si vous souhaitez que chaque consommateur de message non durable puisse extraire des messages de sa propre file d'attente de souscription exclusive, indiquez un nom de file d'attente qui se termine par un astérisque (\*). Ensuite, lorsqu'une application crée un consommateur de message non durable, le client XMS crée une file d'attente dynamique à l'usage exclusif du consommateur de message. Le client XMS utilise la valeur de la propriété pour définir le contenu de la zone **DynamicQName** dans le descripteur d'objet utilisé pour créer la file d'attente dynamique.

La valeur par défaut de la propriété est SYSTEM.JMS.ND.SUBSCRIBER.QUEUE, qui signifie que XMS utilise l'approche de file d'attente partagée par défaut.

Cette propriété est pertinente uniquement dans le domaine publication/abonnement.

***XMSC\_WMQ\_BROKER\_VERSION*****Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory et Destination

**Nom utilisé dans un URI :**

brokerVersion

Type de courtier utilisé par l'application pour une connexion ou pour la destination. Seule une destination représentant une rubrique peut avoir cette propriété.

Les valeurs admises de la propriété sont les suivantes :

**Valeur admise****Explication**

XMSC\_WMQ\_BROKER\_V1

L'application utilise un courtier de publication / abonnement IBM MQ .

L'application peut également utiliser cette valeur si vous migrez depuis la IBM MQ publication / abonnement vers WebSphere Message Broker mais que vous n'avez pas modifié l'application.

XMSC\_WMQ\_BROKER\_V2

L'application utilise un courtier de IBM Integration Bus.

**Valeur admise**

XMSC\_WMQ\_BROKER\_UNSPECIFIED

**Explication**

Lorsque le courtier est migré, définissez cette propriété de sorte que les en-têtes RFH2 ne soient plus utilisés. Après la migration, cette propriété n'est plus significative.

La valeur par défaut pour une fabrique de connexions est XMSC\_WMQ\_BROKER\_UNSPECIFIED, mais par défaut, la propriété n'est pas définie pour une destination. La définition de la propriété pour une destination remplace toute valeur spécifiée par la propriété de fabrique de connexions.

***XMSC\_WMQ\_CCDTURL*****Type de données :**

System.String

**Propriété de :**

ConnectionFactory

**Objets applicables:**

Nom long de l'outil d'administration JMS: CCDTURL

Nom abrégé de l'outil d'administration JMS: CCDT

Adresse URL qui identifie le nom et l'emplacement du fichier contenant la table de définition de canal du client et qui spécifie le mode d'accès au fichier.

Par défaut, cette propriété n'est pas définie.

***XMSC\_WMQ\_CCSID*****Type de données :**

System.Int32

**Propriété de :**

Destination

**Nom utilisé dans un URI :**

CCSID

Identificateur (CCSID) du jeu de caractères codés, ou page de codes, dans lequel se trouvent les chaînes de données de type caractères dans le corps d'un message lorsque le client XMS transmet le message à la destination. S'il est défini pour un message individuel, la propriété JMS\_IBM\_CHARACTER\_SET remplace le CCSID spécifié pour la destination par cette propriété.

La valeur par défaut de la propriété est 1208.

Cette propriété est pertinente pour les messages envoyés à la destination, mais pas pour les messages reçus depuis la destination.

***XMSC\_WMQ\_CHANNEL*****Type de données :**

String

**Propriété de :**

ConnectionFactory

**Objets applicables:**

Nom long de l'outil d'administration JMS: CHANNEL

Nom abrégé de l'outil d'administration JMS: CHAN

Nom du canal à utiliser pour une connexion.

Par défaut, la propriété n'est pas définie.

Cette propriété est pertinente uniquement lorsqu'une application se connecte à un gestionnaire de files d'attente en mode client.



## ***XMSC\_WMQ\_CLIENT\_RECONNECT\_OPTIONS***

### **Type de données :**

String

### **Propriété de :**

ConnectionFactory

### **Objets applicables:**

Nom long de l'outil d'administration JMS: CLIENTRECONNECTOPTIONS

Nom abrégé de l'outil d'administration JMS: CROPT

Cette propriété spécifie les options de reconnexion du client pour les nouvelles connexions créées par cette fabrique. Il se trouve dans XMSC, et est l'un des suivants:

- **WMQ\_CLIENT\_RECONNECT\_AS\_DEF** (par défaut). Utilisez la valeur spécifiée dans le fichier `mqclient.ini`. Définissez la valeur en utilisant la propriété **DefRecon** de la strophe Channels. Il peut s'agir de :
  1. **YES**. Se comporte comme l'option **WMQ\_CLIENT\_RECONNECT**
  2. **NO**. Valeur par défaut. Ne spécifie aucune option de reconnexion
  3. **QMGR**. Se comporte comme l'option **WMQ\_CLIENT\_RECONNECT\_Q\_MGR**
  4. **DISABLED**. Se comporte comme l'option **WMQ\_CLIENT\_RECONNECT\_DISABLED**
- **WMQ\_CLIENT\_RECONNECT**. Se reconnecte à n'importe quel gestionnaire de files d'attente spécifié dans la liste des noms de connexion.
- **WMQ\_CLIENT\_RECONNECT\_Q\_MGR**. Se reconnecte au même gestionnaire de files d'attente avec lequel la connexion d'origine a été établie. **MQRC\_RECONNECT\_QMID\_MISMATCH** est renvoyé si le gestionnaire de files d'attente auquel la tentative de connexion est effectuée (spécifié dans la liste des noms de connexion) a un QMID différent du gestionnaire de files d'attente avec lequel la connexion était établie à l'origine.
- **WMQ\_CLIENT\_RECONNECT\_DISABLED**. La reconnexion est désactivée.

## ***XMSC\_WMQ\_CLIENT\_RECONNECT\_TIMEOUT***

### **Type de données :**

String

### **Propriété de :**

ConnectionFactory

### **Objets applicables:**

Nom long de l'outil d'administration JMS: CLIENTRECONNECTTIMEOUT

Nom abrégé de l'outil d'administration JMS: CRT

La propriété **XMSC\_WMQ\_CLIENT\_RECONNECT\_TIMEOUT** est valide uniquement pour le client XMS .NET géré.

Cette propriété spécifie le temps, en secondes, qu'un connexion client utilise pour essayer de se reconnecter.

Passé ce délai, le client échoue avec **MQRC\_RECONNECT\_FAILED**. Le paramètre par défaut pour cette propriété est **XMSC.WMQ\_CLIENT\_RECONNECT\_TIMEOUT\_DEFAULT**.

La valeur par défaut de cette propriété est 1800.

## ***XMSC\_WMQ\_CONNECTION\_MODE***

### **Type de données :**

System.Int32

### **Propriété de :**

ConnectionFactory

Mode par lequel une application se connecte à un gestionnaire de files d'attente.

Les valeurs admises de la propriété sont les suivantes :

Valeur admise	Explication
XMSC_WMQ_CM_BINDINGS	Connexion à un gestionnaire de files d'attente en mode liaisons, pour une performance optimale. Il s'agit de la valeur par défaut pour C/C++.
XMSC_WMQ_CM_CLIENT	Connexion à un gestionnaire de files d'attente en mode client, pour une gestion totale de pile. Il s'agit de la valeur par défaut pour .NET.
XMSC_WMQ_CM_CLIENT_UNMANAGED (pour .NET uniquement)	Connexion à un gestionnaire de files d'attente qui force une pile de client non géré.

### ***XMSC\_WMQ\_CONNECTION\_NAME\_LIST***

**Type de données :**

String

**Propriété de :**

ConnectionFactory

**Objets applicables:**

Nom long de l'outil d'administration JMS: CONNECTIONNAMELIST

Nom abrégé de l'outil d'administration JMS: CNLIST

Cette propriété spécifie les hôtes auxquels le client tente de se reconnecter après une interruption de sa connexion.

La liste de noms de connexion est une liste séparée par des virgules de paires de port hôte/ip. Le paramètre par défaut pour cette propriété est WMQ\_CONNECTION\_NAME\_LIST\_DEFAULT.

Par exemple, 127.0.0.1(1414) , host2.example.com(1400)

Le paramètre par défaut de cette propriété est localhost(1414).

### ***XMSC\_WMQ\_DUR\_SUBQ***

**Type de données :**

String

**Propriété de :**

Destination

Nom de la file d'attente de souscription pour un abonné durable qui reçoit des messages de la destination. Seule une destination représentant une rubrique peut avoir cette propriété.

Le nom de la file d'attente de souscription doit commencer par les caractères suivants :

SYSTEM.JMS.D.

Si vous voulez que tous les abonnés durables partagent une file d'attente de souscription, spécifiez le nom complet de la file d'attente partagée. Une file d'attente du nom spécifié doit exister avant qu'une application puisse créer un abonné durable.

Si vous souhaitez que chaque abonné durable puisse extraire des messages de sa propre file d'attente de souscription exclusive, indiquez un nom de file d'attente qui se termine par un astérisque (\*). Ensuite, lorsqu'une application crée un abonné durable, le client XMS crée une file d'attente dynamique à l'usage exclusif de l'abonné durable. Le client XMS utilise la valeur de la propriété pour définir le contenu de la zone **DynamicQName** dans le descripteur d'objet utilisé pour créer la file d'attente dynamique.

La valeur par défaut de la propriété est SYSTEM.JMS.D.SUBSCRIBER.QUEUE, qui signifie que XMS utilise l'approche de file d'attente partagée par défaut.

Cette propriété est pertinente uniquement dans le domaine publication/abonnement.

## ***XMSC\_WMQ\_ENCODING***

**Type de données :**

System.Int32

**Propriété de :**

Destination

Mode de représentation des données numériques dans le corps d'un message lorsque le client XMS transmet le message à la destination. S'il est défini pour un message individuel, la propriété [JMS\\_IBM\\_ENCODING](#) remplace le codage spécifié pour la destination par cette propriété. La propriété spécifie la représentation d'entiers binaires, d'entiers décimaux condensés et de nombres à virgule flottante.

Les valeurs valides de la propriété sont les mêmes que celles qui peuvent être spécifiées dans la zone **Encoding** d'un descripteur de message.

Une application peut utiliser les constantes nommées suivantes pour définir la propriété :

<b>Constante nommée</b>	<b>Explication</b>
MQENC_INTEGER_NORMAL	Codage d'entier normal
MQENC_INTEGER_REVERSED	Codage d'entier inversé
MQENC_DECIMAL_NORMAL	Codage décimal condensé normal
MQENC_DECIMAL_REVERSED	Codage décimal condensé inversé
MQENC_FLOAT_IEEE_NORMAL	Codage à virgule flottante IEEE normal
MQENC_FLOAT_IEEE_REVERSED	Codage à virgule flottante IEEE inversé
MQENC_FLOAT_S390	Codage à virgule flottante de l'architecture z/OS
MQENC_NATIVE	Codage machine natif

Pour former une valeur pour la propriété, l'application peut ajouter trois des constantes suivantes :

- Une constante dont le nom commence par MQENC\_INTEGER, pour spécifier la représentation d'entiers binaires
- Une constante dont le nom commence par MQENC\_DECIMAL, pour spécifier la représentation d'entiers décimaux condensés
- Une constante dont le nom commence par MQENC\_FLOAT, pour spécifier la représentation de nombres à virgule flottante

L'application peut également définir la propriété à MQENC\_NATIVE, dont la valeur dépend de l'environnement.

La valeur par défaut de la propriété est MQENC\_NATIVE.

Cette propriété est pertinente pour les messages envoyés à la destination, mais pas pour les messages reçus depuis la destination.

## ***XMSC\_WMQ\_FAIL\_IF QUIESCE***

**Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory et Destination

**Nom utilisé dans un URI :**

failIfQuiesce

**Objets applicables:**

Nom long de l'outil d'administration JMS: FAILIFQUIESCE

Nom abrégé de l'outil d'administration JMS: FIQ

Indique que les appels à certaines méthodes échouent si le gestionnaire de files d'attente auquel l'application est connectée est à l'état de mise au repos.

Les valeurs admises de la propriété sont les suivantes :

Valeur admise	Explication
XMSC_WMQ_FIQ_YES	Les appels à certaines méthodes échouent si le gestionnaire de files d'attente est à l'état de mise au repos. Lorsque l'application détecte que le gestionnaire de files d'attente est au repos, elle termine la tâche immédiate et ferme la connexion, ce qui permet l'arrêt du gestionnaire de files d'attente.
XMSC_WMQ_FIQ_NO	Aucun appel de méthode n'échoue si le gestionnaire de files d'attente est à l'état de mise au repos. Si vous spécifiez cette valeur, l'application ne peut pas détecter que le gestionnaire de files d'attente est à l'état de mise au repos. L'application peut continuer à effectuer des opérations sur le gestionnaire de files d'attente et donc, empêcher l'arrêt de celui-ci.

La valeur par défaut d'une fabrique de connexions est XMSC\_WMQ\_FIQ\_YES mais, aucune destination par défaut n'est définie. La définition de la propriété pour une destination remplace toute valeur spécifiée par la propriété de fabrique de connexions.

**XMSC\_WMQ\_MESSAGE\_BODY****Type de données :**

System.Int32

**Propriété de :**

Destination

Cette propriété détermine si une application XMS traite l' MQRFH2 d'un message IBM MQ dans le cadre de la charge utile du message (c'est-à-dire dans le cadre du corps du message).

**Remarque :** Lors de l'envoi de messages à une destination, la propriété XMSC\_WMQ\_MESSAGE\_BODY remplace la propriété de destination XMS existante XMSC\_WMQ\_TARGET\_CLIENT.

Les valeurs valides de cette propriété sont les suivantes :

**XMSC\_WMQ\_MESSAGE\_BODY\_JMS**

**Réception:** Le type et le corps du message XMS entrant sont déterminés par le contenu de MQRFH2 (s'il est présent) ou de MQMD (s'il n'existe pas de MQRFH2) dans le message IBM MQ reçu.

**Envoi:** Le corps du message XMS sortant contient un en-tête MQRFH2 ajouté en préfixe et généré automatiquement en fonction des propriétés du message XMS et des zones d'en-tête.

**XMSC\_WMQ\_MESSAGE\_BODY\_MQ**

**Réception:** Le type de message XMS entrant est toujours `ByteMessage`, quel que soit le contenu du message IBM MQ reçu ou la zone de format du MQMD reçu. Le corps du message XMS correspond aux données de message non modifiées renvoyées par l'appel API du fournisseur de messagerie sous-jacent. Le jeu de caractères et l'encodage des données dans le corps du message sont déterminés par les zones `CodedCharSetId` et `Encoding fields` du MQMD. Le format des données dans le corps du message est déterminé par la zone `Format field` du MQMD.

**Envoyer:** Le corps du message XMS sortant contient le contenu de l'application tel qu'il est et aucun en-tête IBM MQ généré automatiquement n'est ajouté au corps.

**XMSC\_WMQ\_MESSAGE\_BODY\_UNSPECIFIED**

**Recevoir:** le client XMS détermine une valeur appropriée pour cette propriété. Sur un chemin de réception, cette valeur est celle de la propriété `WMQ_MESSAGE_BODY_JMS`.

**Envoyer:** le client XMS détermine une valeur appropriée pour cette propriété. Sur un chemin d'envoi, cette valeur est celle de la propriété XMSC\_WMQ\_TARGET\_CLIENT.

Par défaut, cette propriété est définie sur XMSC\_WMQ\_MESSAGE\_BODY\_UNSPECIFIED.

### ***XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT***

**Type de données :**

System.Int32

**Propriété de :**

Destination

Détermine le niveau de contexte de message à définir par l'application XMS . L'application doit être exécutée avec les droits appropriés pour que cette propriété soit appliquée.

Les valeurs valides pour cette propriété sont :

**XMSC\_WMQ\_MDCTX\_DEFAULT**

Pour les messages sortants, l'appel API MQOPEN API et la structure MQPMO ne spécifient pas d'options de contexte de message explicites.

**XMSC\_WMQ\_MDCTX\_SET\_IDENTITY\_CONTEXT**

L'appel API MQOPEN API spécifie l'option de contexte de message MQOO\_SET\_IDENTITY\_CONTEXT et la structure MQPMO spécifie MQPMO\_SET\_IDENTITY\_CONTEXT.

**XMSC\_WMQ\_MDCTX\_SET\_ALL\_CONTEXT**

L'appel API MQOPEN API spécifie l'option de contexte de message MQOO\_SET\_ALL\_CONTEXT et la structure MQPMO spécifie MQPMO\_SET\_ALL\_CONTEXT.

Par défaut, cette propriété est définie sur XMSC\_WMQ\_MDCTX\_DEFAULT.

**Remarque :** Cette propriété n'est pas pertinente lorsqu'une application se connecte à WebSphere Application Server service integration bus.

Les propriétés suivantes nécessitent que la propriété XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT soit définie sur la valeur XMSC\_WMQ\_MDCTX\_SET\_IDENTITY\_CONTEXT ou XMSC\_WMQ\_MDCTX\_SET\_ALL\_CONTEXT lors de l'envoi d'un message :

- JMS\_IBM\_MQMD\_USERIDENTIFIER
- JMS\_IBM\_MQMD\_ACCOUNTINGTOKEN
- JMS\_IBM\_MQMD\_APPLIDENTITYDATA

Les propriétés suivantes nécessitent que la propriété XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT soit définie sur XMSC\_WMQ\_MDCTX\_SET\_ALL\_CONTEXT lors de l'envoi d'un message :

- JMS\_IBM\_MQMD\_PUTAPPLTYPE
- JMS\_IBM\_MQMD\_PUTAPPLNAME
- JMS\_IBM\_MQMD\_PUTDATE
- JMS\_IBM\_MQMD\_PUTTIME
- JMS\_IBM\_MQMD\_APPLORIGINDATA

### ***XMSC\_WMQ\_MQMD\_READ\_ENABLED***

**Type de données :**

System.Int32

**Propriété de :**

Destination

Cette propriété détermine si une application XMS peut extraire les valeurs des zones MQMD ou non.

Les valeurs valides pour cette propriété sont :

**XMSC\_WMQ\_READ\_ENABLED\_NO**

Lors de l'envoi de messages, les propriétés JMS\_IBM\_MQMD\* d'un message envoyé ne sont pas mises à jour pour refléter les valeurs de zone mises à jour dans le MQMD.

Lors de la réception de messages, aucune propriété JMS\_IBM\_MQMD\* d'un message reçu n'est disponible, même si elles sont en partie ou toutes définies par l'émetteur.

**XMSC\_WMQ\_READ\_ENABLED\_YES**

Lors de l'envoi de messages, toutes les propriétés JMS\_IBM\_MQMD\* d'un message envoyé sont mises à jour pour refléter les valeurs de zone mises à jour dans le MQMD, y compris les propriétés que l'émetteur n'a pas explicitement définies.

Lors de la réception de messages, toutes les propriétés JMS\_IBM\_MQMD\* d'un message reçu sont disponibles, y compris les propriétés que l'émetteur n'a pas explicitement définies.

Par défaut, cette propriété est définie sur XMSC\_WMQ\_READ\_ENABLED\_NO.

**XMSC\_WMQ\_MQMD\_WRITE\_ENABLED****Type de données :**

System.Int32

**Propriété de :**

Destination

Cette propriété détermine si une application XMS peut ou non définir les valeurs des zones MQMD.

Les valeurs valides pour cette propriété sont :

**XMSC\_WMQ\_WRITE\_ENABLED\_NO**

Toutes les propriétés JMS\_IBM\_MQMD\* sont ignorées et leurs valeurs ne sont pas copiées dans la structure MQMD sous-jacente.

**XMSC\_WMQ\_WRITE\_ENABLED\_YES**

Les propriétés JMS\_IBM\_MQMD\* sont traitées. Leurs valeurs sont copiées dans la structure sous-jacente.

Par défaut, cette propriété est définie sur XMSC\_WMQ\_WRITE\_ENABLED\_NO.

**XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED****Type de données :**

System.Int32

**Propriété de :**

Destination

Cette propriété détermine si les expéditeurs de message sont autorisés à utiliser les insertions asynchrones pour envoyer des messages à cette destination.

Les valeurs valides pour cette propriété sont :

**XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_AS\_DEST**

Détermine si les insertions asynchrones sont autorisées conformément à la définition de rubrique ou de file d'attente.

**XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_AS\_Q\_DEF**

Détermine si les insertions asynchrones sont autorisées conformément à la définition de file d'attente.

**XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_AS\_TOPIC\_DEF**

Détermine si les insertions asynchrones sont autorisées conformément à la définition de rubrique.

**XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_DISABLED**

Les insertions asynchrones ne sont pas autorisées.

### **XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_ENABLED**

Les insertions asynchrones sont autorisées.

Par défaut, cette propriété est définie sur XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_AS\_DEST.

**Remarque :** Cette propriété n'est pas pertinente lorsqu'une application se connecte à WebSphere Application Server service integration bus.

### **XMSC\_WMQ\_READ\_AHEAD\_ALLOWED**

**Type de données :**

System.Int32

**Propriété de :**

Destination

Cette propriété détermine si les consommateurs de message et les navigateurs de files d'attente sont autorisés à utiliser la lecture anticipée pour extraire les messages non permanents et non transactionnels de cette destination dans une mémoire tampon interne avant de les recevoir.

Les valeurs valides pour cette propriété sont :

#### **XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_AS\_Q\_DEF**

Détermine si la lecture anticipée est autorisée conformément à la définition de file d'attente.

#### **XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_AS\_TOPIC\_DEF**

Détermine si la lecture anticipée est autorisée conformément à la définition de rubrique.

#### **XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_AS\_DEST**

Détermine si la lecture anticipée est autorisée conformément à la définition de file d'attente ou de rubrique.

#### **XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_DISABLED**

La lecture anticipée n'est pas autorisé lors de la consommation ou de la consultation de messages.

#### **XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_ENABLED**

La lecture anticipée est autorisée.

Par défaut, cette propriété est définie sur XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_AS\_DEST.

### **XMSC\_WMQ\_READ\_AHEAD\_CLOSE\_POLICY**

**Type de données :**

System.Int32

**Propriété de :**

Destination

Cette propriété détermine, pour les messages distribués à un programme d'écoute de messages asynchrone, ce qu'il advient des messages dans la mémoire tampon interne de lecture anticipée lorsque le consommateur de messages est fermé.

Cette propriété est applicable en spécifiant les options de fermeture de file d'attente lors de la consommation des messages en provenance d'une destination, mais non applicable lors de l'envoi de messages vers une destination.

Cette propriété est ignorée pour les navigateurs de files d'attente, les messages étant toujours disponibles dans la file d'attente pendant l'exploration.

Les valeurs valides pour cette propriété sont :

### **XMSC\_WMQ\_READ\_AHEAD\_CLOSE\_POLICY\_DELIVER\_CURRENT**

Seul l'appel du programme d'écoute des messages se termine avant d'être renvoyé, laissant potentiellement des messages dans la mémoire tampon interne de lecture anticipée qui sont ensuite supprimés.

### **XMSC\_WMQ\_READ\_AHEAD\_CLOSE\_POLICY\_DELIVER\_ALL**

Tous les messages dans la mémoire tampon interne de lecture anticipée sont distribués au programme d'écoute des messages de l'application avant d'être renvoyés.

Par défaut, cette propriété est définie sur `XMSC_WMQ_READ_AHEAD_CLOSE_POLICY_DELIVER_CURRENT`.

#### **Remarque :**

#### **Arrêt anormal de l'application**

Tous les messages de la mémoire tampon de lecture anticipée sont perdus lorsqu'une application XMS s'arrête brutalement.

#### **Implications pour les transactions**

La lecture anticipée est désactivée lorsque les applications utilisent des transactions. L'application ne voit donc aucune différence dans le comportement lorsqu'elles utilisent les sessions transactionnelles.

#### **Implications des modes de reconnaissance de session**

La lecture anticipée est activée sur une session non transactionnelle lorsque le mode d'accusé de réception est `XMSC_AUTO_ACKNOWLEDGE` ou `XMSC_DUPS_OK_ACKNOWLEDGE`. La lecture anticipée est désactivée si le mode d'accusé de réception de la session est `XMSC_CLIENT_ACKNOWLEDGE`, que la session soit transactionnelle ou non.

#### **Implications pour les navigateurs de files d'attente et les sélecteurs de navigateur de files d'attente**

Les navigateurs de file d'attente et les sélecteurs de navigateur de file d'attente, utilisés dans les applications XMS, bénéficient des avantages de la lecture anticipée en termes de performances. La fermeture du navigateur de files d'attente ne dégrade pas les performances car le message est toujours disponible dans la file d'attente pour d'autres opérations. Il n'y a pas d'autres implications pour les navigateurs de file d'attente et les sélecteurs de navigateur de file d'attente en dehors des avantages de la lecture anticipée en termes de performances.

### **XMSC\_WMQ\_HOST\_NAME**

#### **Type de données :**

String

#### **Propriété de :**

ConnectionFactory

#### **Objets applicables:**

Nom long de l'outil d'administration JMS: HOSTNAME

Nom abrégé de l'outil d'administration JMS: HOST

Nom d'hôte ou adresse IP du système sur lequel s'exécute un gestionnaire de files d'attente.

Cette propriété est utilisée uniquement lorsqu'une application se connecte à un gestionnaire de files d'attente en mode client. La propriété est utilisée avec la propriété `XMSC_WMQ_PORT` pour identifier le gestionnaire de files d'attente.

La valeur par défaut de la propriété est `localhost`.

### **XMSC\_WMQ\_LOCAL\_ADDRESS**

#### **Type de données :**

String

#### **Propriété de :**

ConnectionFactory

#### **Objets applicables:**

Nom long de l'outil d'administration JMS: LOCALADDRESS



Nom abrégé de l'outil d'administration JMS: LA

Pour une connexion à un gestionnaire de files d'attente, cette propriété spécifie l'interface de réseau local et/ou le port ou la plage de ports à utiliser.

La valeur de la propriété est une chaîne au format suivant :

`[nom_hôte][(port_bas)[,port_haut]]`

La signification des variables est la suivante :

**nom\_hôte**

Nom d'hôte ou adresse IP de l'interface de réseau local à utiliser pour la connexion.

Cette information est requise uniquement si le système sur lequel l'application s'exécute dispose d'au moins deux interfaces réseau et que vous devez pouvoir spécifier l'interface qui doit être utilisée pour la connexion. Si le système dispose d'une seule interface réseau, seule cette interface peut être utilisée. Si le système dispose d'au moins deux interfaces réseau et que vous ne spécifiez pas quelle interface doit être utilisée, celle-ci est sélectionnée au hasard.

**port\_bas**

Numéro du port local à utiliser pour la connexion.

Si *port\_haut* est aussi spécifiée, *port\_bas* est interprétée comme correspondant au numéro de port le plus bas de la plage de ports.

**port\_haut**

Numéro de port le plus élevé dans la plage de ports. Un des ports de la plage spécifiée doit être utilisé pour la connexion.

La longueur maximale de la chaîne est de 48 caractères.

Voici quelques exemples de valeurs valides pour la propriété :

JUPITER  
9.20.4.98  
JUPITER(1000)  
9.20.4.98(1000,2000)  
(1000)  
(1000,2000)

Par défaut, la propriété n'est pas définie.

Cette propriété est pertinente uniquement lorsqu'une application se connecte à un gestionnaire de files d'attente en mode client.

**XMSC\_WMQ\_MESSAGE\_SELECTION**

**Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory

Détermine si la sélection des messages est effectuée par le client XMS ou par le courtier.

Les valeurs admises de la propriété sont les suivantes :

<b>Valeur admise</b>	<b>Explication</b>
XMSC_WMQ_MSEL_CLIENT	La sélection de message est faite par le client XMS.
XMSC_WMQ_MSEL_BROKER	La sélection de message est faite par le courtier.

La valeur par défaut est XMSC\_WMQ\_MSEL\_CLIENT.

Cette propriété est pertinente uniquement dans le domaine publication/abonnement. La sélection de message par le courtier n'est pas prise en charge si la propriété XMSC\_WMQ\_BROKER\_VERSION est définie sur XMSC\_WMQ\_BROKER\_V1.

### ***XMSC\_WMQ\_MSG\_BATCH\_SIZE***

**Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory

Nombre maximal de messages à extraire d'une file d'attente en un seul lot lorsque la distribution asynchrone de messages est utilisée.

Lorsqu'une application utilise la distribution asynchrone de messages, sous certaines conditions, le client XMS extrait un lot de messages d'une file d'attente avant de réacheminer chaque message de manière individuelle à l'application. Cette propriété spécifie le nombre maximal de messages pouvant constituer un lot.

Il s'agit d'un entier positif ; la valeur par défaut est 10. Envisagez de définir une autre valeur pour cette propriété uniquement si vous devez résoudre un problème spécifique de performance.

Si une application est connectée à un gestionnaire de files d'attente via un réseau, l'augmentation de la valeur de cette propriété peut réduire les surcharges réseau et les temps de réponse, mais augmenter la quantité de mémoire requise pour stocker les messages sur un système client. À l'inverse, la diminution de la valeur de cette propriété peut augmenter les surcharges réseau et les temps de réponse, mais réduire la quantité de mémoire requise pour stocker les messages.

### ***XMSC\_WMQ\_POLLING\_INTERVAL***

**Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory

Si chaque programme d'écoute de message dans une session ne comporte pas de message approprié dans sa file d'attente, cette valeur est l'intervalle de temps maximal, en millisecondes, qui s'écoule avant que chaque programme d'écoute de message essaie à nouveau d'obtenir un message à partir de sa file d'attente.

Si l'absence de message approprié est fréquemment observée pour l'un quelconque des écouteurs de messages au sein d'une session, envisagez d'augmenter la valeur de cette propriété.

La valeur de la propriété est un entier positif. La valeur par défaut est 5000.

### ***XMSC\_WMQ\_PORT***

**Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory

**Objets applicables:**

Nom long de l'outil d'administration JMS: PORT

Nom abrégé de l'outil d'administration JMS: PORT

Numéro du port sur lequel un gestionnaire de files d'attente écoute les demandes entrantes.

Cette propriété est utilisée uniquement lorsqu'une application se connecte à un gestionnaire de files d'attente en mode client. La propriété est utilisée avec la propriété XMSC\_WMQ\_HOST\_NAME pour identifier le gestionnaire de files d'attente.

La valeur par défaut de la propriété est XMSC\_WMQ\_DEFAULT\_CLIENT\_PORT, ou 1414.

## ***XMSC\_WMQ\_PROVIDER\_VERSION***

**Type de données :**

String

**Propriété de :**

ConnectionFactory

Version, édition, niveau de modification et groupe de correctifs du gestionnaire de files d'attente auquel l'application a l'intention de se connecter. Les valeurs valides de cette propriété sont les suivantes :

- Non spécifié

Ou une chaîne dans l'un des formats suivants

- V.R.M.F
- V.R.M
- V.R
- V

où V, R, M et F sont des entiers strictement positifs.

Par défaut, cette propriété n'est pas spécifiée.

La version du client IBM MQ joue également un rôle majeur dans la possibilité pour une application XMS client d'utiliser les fonctions IBM MQ spécifiques.

**Remarque :** Une propriété système XMSC\_WMQ\_OVERRIDEPROVIDERVERSION se substitue à la propriété XMSC\_WMQ\_PROVIDER\_VERSION. Cette propriété peut être utilisée si vous ne pouvez pas modifier le paramètre de fabrication de connexions.

## ***XMSC\_WMQ\_PUB\_ACK\_INTERVAL***

**Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory

Nombre de messages publiés par un diffuseur de publications avant que le client XMS ne demande un accusé de réception du courtier.

Si vous réduisez la valeur de cette propriété, le client demande des accusés de réception plus souvent et, par conséquent, les performances du diffuseur de publications diminuent. Si vous augmentez cette valeur, le client observe un délai plus long avant d'émettre une exception sur un échec du courtier.

La valeur de la propriété est un entier positif. La valeur par défaut est 25.

## ***XMSC\_WMQ\_QMGR\_CCSID***

**Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory

Identificateur (CCSID) du jeu de caractères codés, ou page de codes, dans lequel les zones de données de type caractères définies dans l'interface MQI (Message Queue Interface) sont échangées entre le client XMS et le client IBM MQ . Cette propriété ne s'applique pas pour les chaînes de données de type caractères du corps des messages.

Lorsque l'application XMS se connecte à un gestionnaire de files d'attente en mode client, le client XMS se connecte au client IBM MQ . Les informations échangées entre les deux clients contiennent des zones de données de type caractères définies dans l'interface MQI. Dans des circonstances normales, le client IBM MQ considère que ces zones sont dans la page de codes du système sur lequel les clients sont exécutés.

Si le client XMS envoie et attend de recevoir ces zones dans une page de codes différente, vous devez définir cette propriété pour en informer le client IBM MQ.

Lorsque le client IBM MQ réachemine ces zones de données de type caractères vers le gestionnaire de files d'attente, les données doivent être converties si nécessaire dans la page de codes utilisée par le gestionnaire de files d'attente. De même, lorsque le client IBM MQ reçoit ces zones en provenance du gestionnaire de files d'attente, les données doivent être converties si nécessaire dans la page de codes dans laquelle le client XMS s'attend à les recevoir. Le client IBM MQ utilise cette propriété pour effectuer les conversions de données.

Par défaut, la propriété n'est pas définie.

La définition de cette propriété revient à définir la variable d'environnement MQCCSID pour un client IBM MQ prenant en charge les applications client IBM MQ natives. Pour plus d'informations sur cette variable d'environnement, voir [MQCCSID](#).

### ***XMSC\_WMQ\_QUEUE\_MANAGER***

**Type de données :**

String

**Propriété de :**

ConnectionFactory

**Objets applicables:**

Nom long de l'outil d'administration JMS: QMANAGER

Nom abrégé de l'outil d'administration JMS: QMGR

Nom du gestionnaire de files d'attente auquel se connecter.

Par défaut, la propriété n'est pas définie.

### ***XMSC\_WMQ\_RECEIVE\_CCSID***

Propriété de destination qui définit le CCSID cible pour la conversion des messages du gestionnaire de files d'attente. La valeur est ignorée sauf si XMSC\_WMQ\_RECEIVE\_CONVERSION est défini sur WMQ\_RECEIVE\_CONVERSION\_QMGR.

**Type de données :**

Entier

**Valeur :**

Tout nombre entier positif.

La valeur par défaut est 1208.

La spécification d'une valeur GMO\_CONVERT dans un message est facultative. Si une valeur GMO\_CONVERT est spécifiée, la conversion s'effectue en fonction de cette valeur.

### ***XMSC\_WMQ\_RECEIVE\_CONVERSION***

Propriété de destination qui détermine si la conversion de données sera effectuée par le gestionnaire de files d'attente.

**Type de données :**

Entier

**Valeurs :**

XMSC\_WMQ\_RECEIVE\_CONVERSION\_CLIENT\_MSG (valeur par défaut): effectue la conversion des données uniquement sur le client XMS . La conversion est toujours effectuée avec la page de codes 1208.

XMSC\_WMQ\_RECEIVE\_CONVERSION\_QMGR: effectue une conversion de données sur le gestionnaire de files d'attente avant d'envoyer un message au client XMS .

## ***XMSC\_WMQ\_RECEIVE\_EXIT***

**Type de données :**

String

**Propriété de :**

ConnectionFactory

Identifie un exit de réception de canal à exécuter.

La valeur de la propriété est une chaîne qui identifie un exit de réception de canal et a le format suivant :

**libraryName**(entryPointName)

où

- **libraryName** correspond au chemin d'accès complet du fichier .dll d'exit géré
- **entryPointName** correspond au nom de classe qualifié par l'espace de nom

Exemple: C:\MyReceiveExit.dll(MyReceiveExitNameSpace.MyReceiveExitClassName)

Par défaut, la propriété n'est pas définie.

Cette propriété est pertinente uniquement lorsqu'une application se connecte à un gestionnaire de files d'attente en mode client géré. En outre, seuls les exits gérés sont pris en charge.

## ***XMSC\_WMQ\_RECEIVE\_EXIT\_INIT***

**Type de données :**

String

**Propriété de :**

ConnectionFactory

Données utilisateur transmises à un exit de réception de canal lors de l'appel.

La valeur de la propriété est une chaîne. Par défaut, la propriété n'est pas définie.

Cette propriété est pertinente uniquement lorsqu'une application se connecte à un gestionnaire de files d'attente en mode client géré et que la propriété «[XMSC\\_WMQ\\_RECEIVE\\_EXIT](#)», à la page 2173 est définie.

## ***XMSC\_WMQ\_RESOLVED\_QUEUE\_MANAGER***

**Type de données :**

String

**Propriété de :**

ConnectionFactory

Cette propriété est utilisée pour obtenir le nom du gestionnaire de files d'attente auquel il est connecté.

Lorsqu'il est utilisé avec une table de définition de canal du client, ce nom peut être différent de celui du gestionnaire de files d'attente de la fabrique de connexions.

## ***XMSC\_WMQ\_RESOLVED\_QUEUE\_MANAGER\_ID***

**Type de données :**

String

**Propriété de :**

ConnectionFactory

Cette propriété est remplie avec l'ID du gestionnaire de files d'attente après la connexion.

## ***XMSC\_WMQ\_SECURITY\_EXIT***

**Type de données :**

String

**Propriété de :**

ConnectionFactory

Identifie un exit de sécurité de canal.

La valeur de la propriété est une chaîne qui identifie un exit de sécurité de canal et a le format suivant :

**libraryName**(entryPointName)

où

- **libraryName** correspond au chemin d'accès complet du fichier .dll d'exit géré
- **entryPointName** correspond au nom de classe qualifié par l'espace de nom

Par exemple, C:\MySecurityExit.dll(MySecurityExitNameSpace.MySecurityExitClassName)

La longueur maximale de la chaîne est de 128 caractères.

Par défaut, la propriété n'est pas définie.

Cette propriété est pertinente uniquement lorsqu'une application se connecte à un gestionnaire de files d'attente en mode client géré. En outre, seuls les exits gérés sont pris en charge.

***XMSC\_WMQ\_SECURITY\_EXIT\_INIT*****Type de données :**

String

**Propriété de :**

ConnectionFactory

Données utilisateur transmises à un exit de sécurité de canal lors de l'appel.

La longueur maximale de la chaîne de données utilisateur est de 32 caractères.

Par défaut, la propriété n'est pas définie.

Cette propriété est pertinente uniquement lorsqu'une application se connecte à un gestionnaire de files d'attente en mode client géré et que la propriété [«XMSC\\_WMQ\\_SECURITY\\_EXIT»](#), à la page 2173 est définie.

***XMSC\_WMQ\_SEND\_EXIT*****Type de données :**

String

**Propriété de :**

ConnectionFactory

Identifie un exit d'émission de canal.

La valeur de la propriété est une chaîne. Un exit d'émission de canal a le format suivant :

**libraryName**(entryPointName)

où

- **libraryName** correspond au chemin d'accès complet du fichier .dll d'exit géré
- **entryPointName** correspond au nom de classe qualifié par l'espace de nom

Exemple : C:\MySendExit.dll(MySendExitNameSpace.MySendExitClassName)

Par défaut, la propriété n'est pas définie.

Cette propriété est pertinente uniquement lorsqu'une application se connecte à un gestionnaire de files d'attente en mode client géré. En outre, seuls les exits gérés sont pris en charge.

## ***XMSC\_WMQ\_SEND\_EXIT\_INIT***

### **Type de données :**

String

### **Propriété de :**

ConnectionFactory

Données utilisateur qui sont transmises aux exits d'émission une fois ceux-ci appelés.

La valeur de la propriété est une chaîne de un ou plusieurs éléments de données utilisateur séparés par des virgules. Par défaut, la propriété n'est pas définie.

Les règles de spécification des données utilisateur qui sont transmises à une séquence d'exits d'émission de canal sont les mêmes que les règles de spécification de données utilisateur transmises à une séquence d'exits de réception de canal. Pour les règles, voir [«XMSC\\_WMQ\\_RECEIVE\\_EXIT\\_INIT», à la page 2173.](#)

Cette propriété est pertinente uniquement lorsqu'une application se connecte à un gestionnaire de files d'attente en mode client géré et que la propriété [«XMSC\\_WMQ\\_SEND\\_EXIT», à la page 2174](#) est définie.

## ***XMSC\_WMQ\_SEND\_CHECK\_COUNT***

### **Type de données :**

System.Int32

### **Propriété de :**

ConnectionFactory

Nombre d'appels d'envoi à autoriser entre deux vérifications d'erreurs asynchrones, au cours d'une même session XMS non transactionnelle.

Par défaut, cette propriété est définie sur 0.

## ***XMSC\_WMQ\_SHARE\_CONV\_ALLOWED***

### **Type de données :**

System.Int32

### **Propriété de :**

ConnectionFactory

### **Objets applicables:**

Nom long de l'outil d'administration JMS: SHARECONVALLOWED

Nom abrégé de l'outil d'administration JMS: SCALD

Indique si une connexion client peut partager son socket avec d'autres connexions XMS de niveau supérieur à partir du même processus vers le même gestionnaire de files d'attente, si les définitions de canal correspondent. Cette propriété est fournie pour permettre un isolement complet des connexions dans des sockets distincts, lorsque cela est requis pour des raisons de développement, de maintenance ou de fonctionnement. La définition de cette propriété indique simplement à XMS que le socket sous-jacent doit être partagé. Elle n'indique pas le nombre de connexions partageant un socket. Ceci est déterminé par la valeur SHARECNV, négociée entre IBM MQ Client et IBM MQ Server.

Une application peut utiliser les constantes nommées suivantes pour définir la propriété :

- XMSC\_WMQ\_SHARE\_CONV\_ALLOWED\_FALSE - Les connexions ne partagent pas de socket.
- XMSC\_WMQ\_SHARE\_CONV\_ALLOWED\_TRUE - Les connexions partagent un socket.

Par défaut, cette propriété est définie sur XMSC\_WMQ\_SHARE\_CONV\_ALLOWED\_ENABLED.

Cette propriété est pertinente uniquement lorsqu'une application se connecte à un gestionnaire de files d'attente en mode client.

## ***XMSC\_WMQ\_SSL\_CERT\_STORES***

### **Type de données :**

String

### **Propriété de :**

ConnectionFactory

Emplacements des serveurs contenant les listes de révocation de certificat à utiliser sur une connexion SSL à un gestionnaire de files d'attente.

La valeur de la propriété est une liste de une ou plusieurs adresses URL séparées par des virgules. Chaque adresse URL a le format suivant :

```
[user[/password]@]ldap://[serveraddress][:portnum][, ...]
```

Ce format est compatible avec le format MQJMS de base.

La valeur serveraddress doit être vide. Dans ce cas, XMS utilise la chaîne "localhost".

Exemple :

```
myuser/mypassword@ldap://server1.mycom.com:389
ldap://server1.mycom.com
ldap://
ldap://:389
```

Pour .NET uniquement: depuis IBM MQ 8.0, les connexions gérées à IBM MQ (WMQ\_CM\_CLIENT) et les connexions non gérées à IBM MQ (WMQ\_CM\_CLIENT\_UNMANAGED) prennent en charge les connexions TLS/SSL.

Par défaut, la propriété n'est pas définie.

### **Concepts associés**

[Prise en charge de SSL et TLS pour le client .NET non géré](#)

[Prise en charge de SSL et TLS pour le client .NET géré](#)

## ***XMSC\_WMQ\_SSL\_CIPHER\_SPEC***

### **Type de données :**

String

### **Propriété de :**

ConnectionFactory

Nom de la spécification CipherSpec à utiliser sur une connexion sécurisée vers un gestionnaire de files d'attente.

Les spécifications de chiffrement que vous pouvez utiliser avec le support TLS d'IBM MQ sont répertoriées dans le tableau ci-dessous. Lorsque vous demandez un certificat personnel, vous définissez une taille de clé pour la paire de clé publique et de clé privée. La taille de clé utilisée lors de l'établissement de liaison SSL est celle qui est stockée dans le certificat à moins qu'elle soit déterminée par la spécification CipherSpec, comme indiqué dans le tableau. Par défaut, cette propriété n'est pas définie.

<b>Nom du CipherSpec</b>	<b>Protocole utilisé</b>	<b>Algorithme de hachage</b>	<b>Algorithme de chiffrement</b>	<b>Bits de chiffrement</b>	<b>FIPS <sup>1</sup></b>	<b>Suite B 128 bits</b>	<b>Suite B 192 bits</b>
TLS_RSA_WITH_AES_128_CBC_SHA	TLS 1.0	SHA-1	AES	128	Oui	Non	Non
TLS_RSA_WITH_AES_256_CBC_SHA <sup>2</sup>	TLS 1.0	SHA-1	AES	256	Oui	Non	Non



Nom du CipherSpec	Protocole utilisé	Algorithme de hachage	Algorithme de chiffrement	Bits de chiffrement	FIPS <sup>1</sup>	Suite B 128 bits	Suite B 192 bits
TLS_RSA_WITH_DES_CBC_SHA	TLS 1.0	SHA-1	DES	56	Non	Non	Non
TLS_RSA_WITH_3DES_EDE_CBC_SHA <sup>4</sup>	TLS 1.0	SHA-1	3DES	168	Oui	Non	Non
TLS_RSA_WITH_AES_128_GCM_SHA256	TLS 1.2	SHA-256	AES	128	Oui	Non	Non
TLS_RSA_WITH_AES_256_GCM_SHA384	TLS 1.2	SHA-384	AES	256	Oui	Non	Non
TLS_RSA_WITH_AES_128_CBC_SHA256	TLS 1.2	SHA-256	AES	128	Oui	Non	Non
TLS_RSA_WITH_AES_256_CBC_SHA256	TLS 1.2	SHA-256	AES	256	Oui	Non	Non
ECDHE_ECDSA_RC4_128_SHA256	TLS 1.2	SHA-256	RC4	128	Non	Non	Non
ECDHE_ECDSA_3DES_EDE_CBC_SHA256	TLS 1.2	SHA-256	3DES	168	Oui	Non	Non
ECDHE_RSA_RC4_128_SHA256	TLS 1.2	SHA-256	RC4	128	Non	Non	Non
ECDHE_RSA_3DES_EDE_CBC_SHA256	TLS 1.2	SHA-256	3DES	168	Oui	Non	Non
ECDHE_ECDSA_AES_128_CBC_SHA256	TLS 1.2	SHA-256	AES	128	Oui	Non	Non
ECDHE_ECDSA_AES_256_CBC_SHA384	TLS 1.2	SHA-384	AES	256	Oui	Non	Non
ECDHE_RSA_AES_128_CBC_SHA256	TLS 1.2	SHA-256	AES	128	Oui	Non	Non
ECDHE_RSA_AES_256_CBC_SHA384	TLS 1.2	SHA-384	AES	256	Oui	Non	Non
ECDHE_ECDSA_AES_128_GCM_SHA256	TLS 1.2	SHA-256	AES	128	Oui	Oui	Non
ECDHE_ECDSA_AES_256_GCM_SHA384	TLS 1.2	SHA-384	AES	256	Oui	Non	Oui
ECDHE_RSA_AES_128_GCM_SHA256	TLS 1.2	SHA-256	AES	128	Oui	Non	Non
ECDHE_RSA_AES_256_GCM_SHA384	TLS 1.2	SHA-384	AES	256	Oui	Non	Non
TLS_RSA_WITH_NULL_SHA256	TLS 1.2	SHA-256	Aucun	0	Non	Non	Non
ECDHE_RSA_NULL_SHA256	TLS 1.2	SHA-256	Aucun	0	Non	Non	Non
ECDHE_ECDSA_NULL_SHA256	TLS 1.2	SHA-256	Aucun	0	Non	Non	Non

Nom du CipherSpec	Protocole utilisé	Algorithme de hachage	Algorithme de chiffrement	Bits de chiffrement	FIPS <sup>1</sup>	Suite B 128 bits	Suite B 192 bits
TLS_RSA_WITH_NULL_NULL	TLS 1.2	Aucun	Aucun	0	Non	Non	Non
TLS_RSA_WITH_RC4_128_SHA256	TLS 1.2	SHA-256	RC4	128	Non	Non	Non

**Remarques :**

1. Indique si la spécification CipherSpec est conforme à la norme FIPS (Federal Information Processing Standards) 140-2. Pour obtenir une explication de la norme FIPS et des informations sur la configuration de IBM MQ pour le fonctionnement conforme à la norme FIPS 140-2, voir [Federal Information Processing Standards \(FIPS\)](#).
2. Ce CipherSpec ne peut pas être utilisé pour sécuriser une connexion entre IBM MQ Explorer et un gestionnaire de files d'attente, sauf si les fichiers de règles sans restriction appropriés sont appliqués à l'environnement d'exécution Java utilisé par IBM MQ Explorer.
3. Ce CipherSpec a été certifié FIPS 140-2 avant le 19 mai 2007.
4. Lorsqu'IBM MQ est configuré pour un fonctionnement compatible avec la norme FIPS 140-2, cette spécification CipherSpec peut être utilisée pour transférer jusqu'à 32 Go de données avant que la connexion ne s'arrête avec l'erreur AMQ9288. Pour éviter cette erreur, n'utilisez pas la norme DES triple (qui est dépréciée) ou activez la réinitialisation de clé confidentielle lors de l'utilisation de ce CipherSpec dans une configuration FIPS 140-2.

**Concepts associés**

[Intégrité des données de messages](#)

**Tâches associées**

[Sécurisation](#)

[Définition des spécifications CipherSpec](#)

***XMSC\_WMQ\_SSL\_CIPHER\_SUITE***

**Type de données :**

String

**Propriété de :**

ConnectionFactory

Nom de la suite de chiffrement à utiliser sur une connexion TLS à un gestionnaire de files d'attente. Le protocole utilisé lors de la négociation de la connexion sécurisée dépend de la suite de chiffrement spécifiée.

Cette propriété a les valeurs canoniques suivantes :

- SSL\_RSA\_WITH\_DES\_CBC\_SHA
- SSL\_RSA\_EXPORT1024\_WITH\_DES\_CBC\_SHA
- SSL\_RSA\_EXPORT1024\_WITH\_RC4\_56\_SHA
- SSL\_RSA\_EXPORT\_WITH\_RC4\_40\_MD5
- SSL\_RSA\_WITH\_RC4\_128\_MD5
- SSL\_RSA\_WITH\_RC4\_128\_SHA
- SSL\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA
- SSL\_RSA\_WITH\_AES\_128\_CBC\_SHA
- SSL\_RSA\_WITH\_AES\_256\_CBC\_SHA
- SSL\_RSA\_WITH\_DES\_CBC\_SHA

- `SSL_RSA_WITH_3DES_EDE_CBC_SHA`

Cette valeur peut être fournie comme alternative à `XMSC_WMQ_SSL_CIPHER_SPEC`.

Si une valeur non vide est spécifiée pour `XMSC_WMQ_SSL_CIPHER_SPEC`, elle remplace le paramétrage pour `XMSC_WMQ_SSL_CIPHER_SUITE`. Si `XMSC_WMQ_SSL_CIPHER_SPEC` n'a pas de valeur, la valeur de `XMSC_WMQ_SSL_CIPHER_SUITE` est utilisée comme suite de chiffrement à attribuer à IBM Global Security Kit (GSKit). Dans ce cas, la valeur est mappée sur la valeur CipherSpec équivalente, comme décrit dans les mappages de noms CipherSuite et CipherSpec pour les connexions XMS à un IBM MQ gestionnaire de files d'attente.

Si `XMSC_WMQ_SSL_CIPHER_SPEC` et `XMSC_WMQ_SSL_CIPHER_SUITE` sont vides, la zone `pChDef->SSLCipherSpec` est complétée avec des espaces.

Pour .NET uniquement: depuis IBM MQ 8.0, les connexions gérées à IBM MQ (`WMQ_CM_CLIENT`) et les connexions non gérées à IBM MQ (`WMQ_CM_CLIENT_UNMANAGED`) prennent en charge les connexions TLS/SSL.

Par défaut, la propriété n'est pas définie.

#### **Concepts associés**

Prise en charge de SSL et TLS pour le client .NET non géré

Prise en charge de SSL et TLS pour le client .NET géré

### ***XMSC\_WMQ\_SSL\_CRYPTO\_HW***

#### **Type de données :**

String

#### **Propriété de :**

ConnectionFactory

Détails de configuration pour le matériel de cryptographie connecté au système client.

Cette propriété a les valeurs canoniques suivantes :

- `GSK_ACCELERATOR_RAINBOW_CS_OFF`
- `GSK_ACCELERATOR_RAINBOW_CS_ON`
- `GSK_ACCELERATOR_NCIPHER_NF_OFF`
- `GSK_ACCELERATOR_NCIPHER_NF_ON`

Il existe un format spécial pour le matériel de cryptographie PKCS11 (où `DriverPath`, `TokenLabel` et `TokenPassword` sont des chaînes spécifiées par l'utilisateur) :

```
GSK_PKCS11=PKCS#11 DriverPath; PKCS#11 TokenLabel;PKCS#11 TokenPassword
```

XMS n'interprète ni ne modifie le contenu de la chaîne. Il copie la valeur fournie, dans la limite de 256 caractères à un octet, dans la zone `MQSCO.CryptoHardware`.

Pour .NET uniquement: depuis IBM MQ 8.0, les connexions gérées à IBM MQ (`WMQ_CM_CLIENT`) et les connexions non gérées à IBM MQ (`WMQ_CM_CLIENT_UNMANAGED`) prennent en charge les connexions TLS/SSL.

Par défaut, la propriété n'est pas définie.

#### **Concepts associés**

Prise en charge de SSL et TLS pour le client .NET non géré

Prise en charge de SSL et TLS pour le client .NET géré

### ***XMSC\_WMQ\_SSL\_FIPS\_REQUIRED***

#### **Type de données :**

Boolean

**Propriété de :**

ConnectionFactory

La valeur de cette propriété détermine si une application peut ou non utiliser des suites de chiffrement conformes non FIPS. Si cette propriété est pour valeur true, seuls les algorithmes FIPS sont utilisés pour la connexion client/serveur.

Cette propriété peut avoir les valeurs suivantes, qui correspondent aux deux valeurs canoniques pour MQSCO.FlipsRequired :

<i>Tableau 880. Tableau de valeurs pour la propriété MQSCO.FlipsRequired</i>		
<b>Valeur</b>	<b>Description</b>	<b>Valeur correspondante de MQSSL.FlipsRequired</b>
false	Tout CipherSpec peut être utilisé.	MQSSL_FIPS_NO (valeur par défaut)
Oui	Seuls les algorithmes de cryptographie certifiés FIPS peuvent être utilisés dans le CipherSpec s'appliquant à cette connexion client.	MQSSL_FIPS_YES

XMS copie la valeur appropriée dans MQSCO.FlipsRequired avant d'appeler MQCONN.

Pour .NET uniquement: depuis IBM MQ 8.0, les connexions gérées à IBM MQ (WMQ\_CM\_CLIENT) et les connexions non gérées à IBM MQ (WMQ\_CM\_CLIENT\_UNMANAGED) prennent en charge les connexions TLS/SSL.

**Concepts associés**

[Prise en charge de SSL et TLS pour le client .NET non géré](#)

[Prise en charge de SSL et TLS pour le client .NET géré](#)

***XMSC\_WMQ\_SSL\_KEY\_REPOSITORY*****Type de données :**

String

**Propriété de :**

ConnectionFactory

Emplacement du fichier de clés dans lequel les clés et les certificats sont stockés.

XMS copie la chaîne, jusqu'à une limite de 256 caractères à un octet, dans la zone MQSCO.KeyRepository. IBM MQ interprète cette chaîne comme un nom de fichier, chemin d'accès complet compris.

Pour .NET uniquement: depuis IBM MQ 8.0, les connexions gérées à IBM MQ (WMQ\_CM\_CLIENT) et les connexions non gérées à IBM MQ (WMQ\_CM\_CLIENT\_UNMANAGED) prennent en charge les connexions TLS/SSL.

Par défaut, la propriété n'est pas définie.

**Concepts associés**

[Prise en charge de SSL et TLS pour le client .NET non géré](#)

[Prise en charge de SSL et TLS pour le client .NET géré](#)

***XMSC\_WMQ\_SSL\_KEY\_RESETCOUNT*****Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory

Nombre total d'octets non chiffrés envoyés et reçus dans une conversation SSL avant renégociation de la clé confidentielle. Le nombre d'octets inclut les informations de contrôle envoyées par l'agent MCA.

XMS copie la valeur que vous indiquez pour cette propriété dans MQSCO.KeyResetCount avant d'appeler MQCONNX.

Le paramètre MQSCO.KeyRestCount est disponible uniquement à partir de IBM MQ version 6. Si vous définissez cette propriété dans IBM MQ version 5.3, XMS n'essaie pas d'établir la connexion avec le gestionnaire de files d'attente, mais envoie une exception appropriée.

Pour .NET uniquement: depuis IBM MQ 8.0, les connexions gérées à IBM MQ (WMQ\_CM\_CLIENT) et les connexions non gérées à IBM MQ (WMQ\_CM\_CLIENT\_UNMANAGED) prennent en charge les connexions TLS/SSL.

La valeur par défaut de cette propriété est 0, ce qui signifie que les clés confidentielles ne sont jamais renégociées.

### **Concepts associés**

[Prise en charge de SSL et TLS pour le client .NET non géré](#)

[Prise en charge de SSL et TLS pour le client .NET géré](#)

## ***XMSC\_WMQ\_SSL\_PEER\_NAME***

### **Type de données :**

String

### **Propriété de :**

ConnectionFactory

Nom d'homologue à utiliser sur une connexion SSL vers un gestionnaire de files d'attente.

Il n'existe pas de liste de valeurs canoniques pour cette propriété. A la place, vous devez générer cette chaîne conformément aux règles pour SSLPEER.

Exemple de nom d'homologue :

```
"CN=John Smith, O=IBM ,OU=Test , C=GB"
```

XMS copie la chaîne dans la page de codes à un octet requise, puis positionne les valeurs dans MQCD.SSLPeerNamePtr et MQCD.SSLPeerNameLength avant d'appeler MQCONNX.

Cette propriété est pertinente uniquement si l'application se connecte à un gestionnaire de files d'attente en mode client.

Pour .NET uniquement: depuis IBM MQ 8.0, les connexions gérées à IBM MQ (WMQ\_CM\_CLIENT) et les connexions non gérées à IBM MQ (WMQ\_CM\_CLIENT\_UNMANAGED) prennent en charge les connexions TLS/SSL.

Par défaut, la propriété n'est pas définie.

### **Concepts associés**

[Prise en charge de SSL et TLS pour le client .NET non géré](#)

[Prise en charge de SSL et TLS pour le client .NET géré](#)

### **Référence associée**

[SSLPEERNAME](#)

## ***XMSC\_WMQ\_SYNCPOINT\_ALL\_GETS***

### **Type de données :**

System.Boolean

### **Propriété de :**

ConnectionFactory

Indique si tous les messages doivent être extraits des files d'attente sous le contrôle d'un point de synchronisation.

Les valeurs admises de la propriété sont les suivantes :

**Valeur admise**

false

Oui

**Explication**

Lorsque les circonstances le permettent, le client XMS peut extraire les messages des files d'attente sans le contrôle d'un point de synchronisation.

Le client XMS doit extraire tous les messages des files d'attente sous le contrôle d'un point de synchronisation.

La valeur par défaut est false.

***XMSC\_WMQ\_TARGET\_CLIENT*****Type de données :**

System.Int32

**Propriété de :**

Destination

**Nom utilisé dans un URI :**

targetClient

Indique si les messages envoyés à la destination contiennent un en-tête MQRFH2.

Si une application envoie un message contenant un en-tête MQRFH2, l'application destinataire doit être capable de gérer cet en-tête.

Les valeurs admises de la propriété sont les suivantes :

**Valeur admise**

XMSC\_WMQ\_TARGET\_DEST\_JMS

**Explication**

Les messages envoyés à la destination contiennent un en-tête MQRFH2. Indiquez cette valeur si l'application envoie les messages à une autre application XMS , à une application IBM MQ classes for JMS ou à une application IBM MQ native conçue pour gérer un en-tête MQRFH2 .

XMSC\_WMQ\_TARGET\_DEST\_MQ

Les messages envoyés à la destination ne contiennent pas d'en-tête MQRFH2. Spécifiez cette valeur si l'application envoie les messages à une application IBM MQ native qui n'est pas conçue pour gérer les en-têtes MQRFH2.

La valeur par défaut est XMSC\_WMQ\_TARGET\_DEST\_JMS.

***XMSC\_WMQ\_TEMP\_Q\_PREFIX*****Type de données :**

String

**Propriété de :**

ConnectionFactory

Préfixe utilisé pour former le nom de la file d'attente dynamique IBM MQ créée lorsque l'application crée une file d'attente temporaire XMS .

Les règles de formation du préfixe sont les mêmes que les règles de formation du contenu de la zone **DynamicQName** dans un descripteur d'objet, mais le dernier caractère non blanc doit être un astérisque (\*). Si la propriété n'est pas définie, la valeur utilisée est CSQ . \* sur z/OS et AMQ . \* sur les autres plateformes. Par défaut, la propriété n'est pas définie.

Cette propriété est pertinente uniquement dans le domaine point à point.

***XMSC\_WMQ\_TEMP\_TOPIC\_PREFIX*****Type de données :**

String

**Propriété de :**

ConnectionFactory, Destination

Lors de la création de rubriques temporaires, XMS génère une chaîne de rubrique au format "TEMP/TEMPTOPICPREFIX/unique\_id", ou si cette propriété contient la valeur par défaut, cette chaîne, "TEMP/unique\_id", est générée. La spécification d'une valeur non vide permet la définition de files d'attente modèle spécifiques pour la création des files d'attente gérées pour les abonnés à des rubriques temporaires créées sous cette connexion.

Toute chaîne non nulle constituée uniquement de caractères valides pour une chaîne de rubrique IBM MQ est une valeur valide pour cette propriété.

Par défaut, cette propriété est définie sur "" (chaîne vide).

**Remarque :** Cette propriété est pertinente uniquement dans le domaine publication/abonnement.

***XMSC\_WMQ\_TEMPORARY\_MODEL*****Type de données :**

String

**Propriété de :**

ConnectionFactory

Nom de la file d'attente modèle IBM MQ à partir de laquelle une file d'attente dynamique est créée lorsque l'application crée une file d'attente temporaire XMS .

La valeur par défaut de la propriété est SYSTEM.DEFAULT.MODEL.QUEUE.

Cette propriété est pertinente uniquement dans le domaine point à point.

***XMSC\_WMQ\_WILDCARD\_FORMAT*****Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory, Destination

Cette propriété détermine la version de syntaxe des caractères génériques à utiliser.

Lors de l'utilisation de la fonction de publication / abonnement avec IBM MQ '\*'et?' sont traités comme des caractères génériques. Les caractères # et + sont traités comme des caractères génériques lorsque vous utilisez le mode publication/abonnement avec IBM Integration Bus. Cette propriété remplace la propriété XMSC\_WMQ\_BROKER\_VERSION.

Les valeurs valides pour cette propriété sont :

***XMSC\_WMQ\_WILDCARD\_TOPIC\_ONLY***

Reconnaît uniquement les caractères génériques de niveau rubrique, c'est-à-dire '#' et '+' sont traités comme des caractères génériques. Cette valeur est identique à XMSC\_WMQ\_BROKER\_V2.

***XMSC\_WMQ\_WILDCARD\_CHAR\_ONLY***

Reconnaît uniquement les caractères génériques, c'est-à-dire '\*' et '?' sont traités comme des caractères génériques. Cette valeur est identique à XMSC\_WMQ\_BROKER\_V1.

Par défaut, cette propriété est définie sur XMSC\_WMQ\_WILDCARD\_TOPIC\_ONLY.

***XMSC\_WPM\_BUS\_NAME*****Type de données :**

String

**Propriété de :**

ConnectionFactory et Destination

**Nom utilisé dans un URI :**

busName

Pour une fabrique de connexions, nom du bus d'intégration de services auquel l'application se connecte ; pour une destination, nom du bus d'intégration de services dans lequel la destination existe.

Pour une destination sous forme de rubrique, cette propriété est le nom du bus d'intégration de services dans lequel existe l'espace de rubrique associé. Cet espace de rubrique est spécifié par la propriété `XMSC_WPM_TOPIC_SPACE`.

Si la propriété n'est pas définie pour une destination, la file d'attente ou l'espace de rubrique associé est supposé exister dans le bus d'intégration de services auquel l'application se connecte.

Par défaut, la propriété n'est pas définie.

## ***XMSC\_WPM\_CONNECTION\_PROTOCOL***

### **Type de données :**

System.Int32

### **Propriété de :**

Connexion

Protocole de communication utilisé pour la connexion au moteur de messagerie. Cette propriété est en lecture seule.

Les valeurs possibles sont les suivantes :

<b>Valeur</b>	<b>Explication</b>
XMSC_WPM_CP_HTTP	La connexion utilise HTTP sur TCP/IP.
XMSC_WPM_CP_TCP	La connexion utilise TCP/IP.

## ***XMSC\_WPM\_CONNECTION\_PROXIMITY***

### **Type de données :**

System.Int32

### **Propriété de :**

ConnectionFactory

Paramètre de proximité de connexion pour la connexion. Cette propriété détermine quelle doit être la proximité entre le moteur de messagerie auquel l'application se connecte et le serveur d'amorçage.

Les valeurs admises de la propriété sont les suivantes :

<b>Valeur admise</b>	<b>Paramètre de proximité de connexion</b>
XMSC_WPM_CONNECTION_PROXIMITY_BUS	Bus
XMSC_WPM_CONNECTION_PROXIMITY_CLUSTER	Cluster
XMSC_WPM_CONNECTION_PROXIMITY_HOST	Hôte
XMSC_WPM_CONNECTION_PROXIMITY_SERVER	serveur

La valeur par défaut est `XMSC_WPM_CONNECTION_PROXIMITY_BUS`.

## ***XMSC\_WPM\_DUR\_SUB\_HOME***

### **Type de données :**

String

### **Propriété de :**

ConnectionFactory

### **Nom utilisé dans un URI :**

durableSubscriptionHome



Nom du moteur de messagerie où sont gérés les abonnements durables pour une connexion ou une destination. Les messages devant être distribués aux abonnés durables sont stockés sur le point de publication du même moteur de messagerie.

Un accueil d'abonnement durable doit être spécifié pour une connexion avant qu'une application puisse créer un abonné durable qui utilise la connexion. Toute valeur spécifiée pour une destination remplace la valeur spécifiée pour la connexion.

Par défaut, la propriété n'est pas définie.

Cette propriété est pertinente uniquement dans le domaine publication/abonnement.

### ***XMSC\_WPM\_HOST\_NAME***

**Type de données :**

String

**Propriété de :**

Connexion

Nom hôte ou adresse IP du système qui contient le moteur de messagerie auquel l'application est connectée. Cette propriété est en lecture seule.

### ***XMSC\_WPM\_LOCAL\_ADDRESS***

**Type de données :**

String

**Propriété de :**

ConnectionFactory

Pour une connexion à un bus d'intégration de services, cette propriété spécifie l'interface de réseau local et/ou le port ou la plage de ports à utiliser.

La valeur de la propriété est une chaîne au format suivant :

*[nom\_hôte][(port\_bas)[,port\_haut]]*

La signification des variables est la suivante :

***nom\_hôte***

Nom d'hôte ou adresse IP de l'interface de réseau local à utiliser pour la connexion.

Cette information est requise uniquement si le système sur lequel l'application s'exécute dispose d'au moins deux interfaces réseau et que vous devez pouvoir spécifier l'interface qui doit être utilisée pour la connexion. Si le système dispose d'une seule interface réseau, seule cette interface peut être utilisée. Si le système dispose d'au moins deux interfaces réseau et que vous ne spécifiez pas quelle interface doit être utilisée, celle-ci est sélectionnée au hasard.

***port\_bas***

Numéro du port local à utiliser pour la connexion.

Si *port\_haut* est aussi spécifiée, *port\_bas* est interprétée comme correspondant au numéro de port le plus bas de la plage de ports.

***port\_haut***

Numéro de port le plus élevé dans la plage de ports. Un des ports de la plage spécifiée doit être utilisé pour la connexion.

Voici quelques exemples de valeurs valides pour la propriété :

JUPITER

9.20.4.98

JUPITER(1000)

9.20.4.98(1000,2000)

(1000)

(1000,2000)

Par défaut, la propriété n'est pas définie.

### ***XMSC\_WPM\_ME\_NAME***

**Type de données :**

String

**Propriété de :**

Connexion

Nom du moteur de messagerie auquel l'application est connectée. Cette propriété est en lecture seule.

### ***XMSC\_WPM\_NON\_PERSISTENT\_MAP***

**Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory

Niveau de fiabilité des messages non persistants envoyés à l'aide de la connexion.

Les valeurs admises de la propriété sont les suivantes :

**Valeur admise**

XMSC\_WPM\_MAPPING\_AS\_DESTINATION

XMSC\_WPM\_MAPPING\_BEST\_EFFORT\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_EXPRESS\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_RELIABLE\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_RELIABLE\_PERSISTENT

XMSC\_WPM\_MAPPING\_ASSURED\_PERSISTENT

**Niveau de fiabilité**

Déterminé par le niveau de fiabilité par défaut spécifié pour la file d'attente ou l'espace de rubrique dans le bus d'intégration de services

Non persistant - Meilleur effort

Non persistant - Express

Non persistant - Fiable

Persistant - Fiable

Persistant - Assuré

La valeur par défaut est XMSC\_WPM\_MAPPING\_EXPRESS\_NON\_PERSISTENT.

### ***XMSC\_WPM\_PERSISTENT\_MAP***

**Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory

Niveau de fiabilité des messages non persistants envoyés à l'aide de la connexion.

Les valeurs admises de la propriété sont les suivantes :

## Valeur admise

XMSC\_WPM\_MAPPING\_AS\_DESTINATION

XMSC\_WPM\_MAPPING\_BEST\_EFFORT\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_EXPRESS\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_RELIABLE\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_RELIABLE\_PERSISTENT

XMSC\_WPM\_MAPPING\_ASSURED\_PERSISTENT

## Niveau de fiabilité

Déterminé par le niveau de fiabilité par défaut spécifié pour la file d'attente ou l'espace de rubrique dans le bus d'intégration de services

Non persistant - Meilleur effort

Non persistant - Express

Non persistant - Fiable

Persistant - Fiable

Persistant - Assuré

La valeur par défaut est XMSC\_WPM\_MAPPING\_RELIABLE\_PERSISTENT.

## ***XMSC\_WPM\_PORT***

### **Type de données :**

System.Int32

### **Propriété de :**

Connexion

Numéro du port d'écoute du moteur de messagerie auquel l'application est connectée. Cette propriété est en lecture seule.

## ***XMSC\_WPM\_PROVIDER\_ENDPOINTS***

### **Type de données :**

String

### **Propriété de :**

ConnectionFactory

Séquence d'une ou plusieurs adresses de noeud final de serveurs d'amorçage. Les adresses de noeud final sont séparées par des virgules.

Un serveur d'amorçage est un serveur d'applications responsable de la sélection du moteur de messagerie auquel l'application se connecte. L'adresse de noeud final du serveur d'amorçage se présente sous la forme suivante :

*nom\_hôte:numéro\_port:nom\_chaîne*

La signification des composants d'une adresse de noeud final est la suivante :

### ***nom\_hôte***

Nom d'hôte ou adresse IP du système sur lequel se trouve le serveur d'amorçage. Si vous ne spécifiez aucun nom d'hôte ou aucune adresse IP, la valeur par défaut est localhost.

### ***numéro\_port***

Numéro du port sur lequel le serveur d'amorçage est à l'écoute des demandes entrantes. Si vous ne spécifiez aucun numéro de port, la valeur par défaut est 7276.

### ***nom\_chaîne***

Nom d'une chaîne de transport d'amorçage utilisée par le serveur d'amorçage. Les valeurs admises sont les suivantes :

Valeur admise	Nom de la chaîne de transport d'amorçage
XMSC_WPM_BOOTSTRAP_HTTP	BootstrapTunneledMessaging
XMSC_WPM_BOOTSTRAP_HTTPS	BootstrapTunneledSecureMessaging
XMSC_WPM_BOOTSTRAP_SSL	BootstrapSecureMessaging
XMSC_WPM_BOOTSTRAP_TCP	BootstrapBasicMessaging

Si vous ne spécifiez aucun nom la valeur par défaut est XMSC\_WPM\_BOOTSTRAP\_TCP.

Si vous ne spécifiez aucune adresse de noeud final, la valeur par défaut est localhost:7276:BootstrapBasicMessaging.

## ***XMSC\_WPM\_SSL\_CIPHER\_SUITE***

### **Type de données :**

String

### **Propriété de :**

ConnectionFactory

Nom de la CipherSuite à utiliser sur une connexion TLS à un moteur de messagerie WebSphere Application Server service integration bus . Le protocole utilisé lors de la négociation de la connexion sécurisée dépend de la suite de chiffrement spécifiée.

*Tableau 881. Options CipherSuite pour la connexion à un moteur de messagerie WebSphere Application Server service integration bus*

Algorithme de cryptographie	Protocole utilisé
TLS_RSA_WITH_DES_CBC_SHA	TLSv1
TLS_RSA_WITH_3DES_EDE_CBC_SHA	TLSv1
TLS_RSA_WITH_AES_128_CBC_SHA	TLSv1
TLS_RSA_WITH_AES_256_CBC_SHA	TLSv1

### **Remarques :**

- Windows** Les spécifications TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA et TLS\_RSA\_WITH\_AES\_256\_CBC\_SHA CipherSuites sont prises en charge uniquement sur Windows. (Ceci est dicté par GSKit.)
- Deprecated** TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA a été déprécié. Toutefois, vous pouvez tout de même l'utiliser pour transférer jusqu'à 32 Go de données avant que la connexion ne soit arrêtée avec l'erreur AMQ9288. Pour éviter cette erreur, n'utilisez pas la norme DES triple ou activez la réinitialisation de clé confidentielle lors de l'utilisation de ce CipherSpec.

Cette propriété n'a pas de valeur par défaut. Si vous voulez utiliser SSL ou TLS, vous devez spécifier une valeur pour cette propriété, de sorte que votre application puisse se connecter au serveur.

## ***XMSC\_WPM\_SSL\_FIPS\_REQUIRED***

**Remarque :** Sous AIX, Linux, and Windows, IBM MQ fournit la conformité à la norme FIPS 140-2 via le module cryptographique IBM Crypto for C (ICC) . Le certificat de ce module a été déplacé vers le statut Historique. Les clients doivent afficher le certificat IBM Crypto for C (ICC) et prendre connaissance des conseils fournis par le NIST. Un module FIPS 140-3 de remplacement est actuellement en cours et son statut peut être affiché en le recherchant dans la liste des modules NIST CMVP en cours de traitement.

### **Type de données :**

Boolean

**Propriété de :**

ConnectionFactory

La valeur de cette propriété détermine si une application peut ou non utiliser des suites de chiffrement conformes non FIPS. Si cette propriété est définie à true, seuls des algorithmes FIPS sont utilisés pour la connexion client-serveur. La définition de la valeur de la propriété à TRUE empêche l'application d'utiliser des algorithmes de cryptographie conformes non FIPS.

Par défaut, la propriété est paramétrée sur FALSE (c'est-à-dire, avec le mode FIPS désactivé).

***XMSC\_WPM\_SSL\_KEY\_REPOSITORY*****Type de données :**

String

**Propriété de :**

ConnectionFactory

Chemin d'accès au fichier de clés contenant les clés publiques ou privées à utiliser dans la connexion sécurisée.

La définition de la propriété de fichier de clés à la valeur spéciale de `XMSC_WPM_SSL_MS_CERTIFICATE_STORE` spécifie l'utilisation de la base de données Microsoft Windows. L'utilisation de la base de données de clés Microsoft Windows, disponible à partir de **Panneau de configuration > Options Internet > Contenu > Certificats**, dispense du besoin d'une base de données de clés distincte. L'utilisation de cette constante sur Windows x64 et d'autres plateformes n'est pas autorisée.

Par défaut, la propriété n'est pas définie.

***XMSC\_WPM\_SSL\_KEYRING\_LABEL*****Type de données :**

String

**Propriété de :**

ConnectionFactory

Certificat à utiliser lors de l'authentification avec le serveur. Si aucune valeur n'est spécifiée, le certificat par défaut est utilisé.

Par défaut, la propriété n'est pas définie.

***XMSC\_WPM\_SSL\_KEYRING\_PW*****Type de données :**

String

**Propriété de :**

ConnectionFactory

Mot de passe pour le fichier de clés.

Cette propriété peut être une alternative à `XMSC_WPM_SSL_KEYRING_STASH_FILE` pour configurer le mot de passe pour le fichier de clés.

Par défaut, la propriété n'est pas définie.

***XMSC\_WPM\_SSL\_KEYRING\_STASH\_FILE*****Type de données :**

String

**Propriété de :**

ConnectionFactory

Nom d'un fichier binaire contenant le mot de passe du fichier référentiel principal.

Cette propriété peut être une alternative à `XMSC_WPM_SSL_KEYRING_PW` pour configurer le mot de passe pour le fichier de clés.

Par défaut, la propriété n'est pas définie.

### ***XMSC\_WPM\_TARGET\_GROUP***

**Type de données :**

String

**Propriété de :**

ConnectionFactory

Nom d'un groupe cible de moteurs de messagerie. La nature du groupe cible est déterminée par la propriété `XMSC_WPM_TARGET_TYPE`.

Définissez cette propriété si vous voulez restreindre la recherche d'un moteur de messagerie à un sous-groupe de moteurs du bus d'intégration de services. Si vous souhaitez que votre application puisse se connecter à tout moteur de messagerie du bus d'intégration de services, ne définissez pas cette propriété.

Par défaut, la propriété n'est pas définie.

### ***XMSC\_WPM\_TARGET\_SIGNIFICANCE***

**Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory

Signification du groupe cible de moteurs de messagerie.

Les valeurs admises de la propriété sont les suivantes :

<b>Valeur admise</b>	<b>Explication</b>
<code>XMSC_WPM_TARGET_SIGNIFICANCE_PREFERRED</code>	Un moteur de messagerie du groupe cible est sélectionné, si l'un d'entre eux est disponible. Dans le cas contraire, un moteur de messagerie hors du groupe cible est sélectionné, sous réserve qu'il se trouve dans le même bus d'intégration de services.
<code>XMSC_WPM_TARGET_SIGNIFICANCE_REQUIRED</code>	Le moteur de messagerie sélectionné doit être dans le groupe cible. Si aucun moteur de messagerie du groupe cible n'est disponible, le processus de connexion échoue.

La valeur par défaut de la propriété est `XMSC_WPM_TARGET_SIGNIFICANCE_PREFERRED`.

### ***XMSC\_WPM\_TARGET\_TRANSPORT\_CHAIN***

**Type de données :**

String

**Propriété de :**

ConnectionFactory

Nom de la chaîne de transport entrant que l'application doit utiliser pour se connecter à un moteur de messagerie.

La valeur de la propriété peut être le nom d'une chaîne de transport entrant disponible dans le serveur d'applications qui héberge le moteur de messagerie. La constante nommée suivante est fournie pour une des chaînes de transport entrant prédéfinies :

**Constante nommée**

XMSC\_WPM\_TARGET\_TRANSPORT\_CHAIN\_BASIC

**Nom de la chaîne de transport**

InboundBasicMessaging

La valeur par défaut de la propriété est XMSC\_WPM\_TARGET\_TRANSPORT\_CHAIN\_BASIC.

***XMSC\_WPM\_TARGET\_TYPE*****Type de données :**

System.Int32

**Propriété de :**

ConnectionFactory

Type du groupe cible de moteurs de messagerie. Cette propriété détermine la nature du groupe cible identifié par la propriété XMSC\_WPM\_TARGET\_GROUP.

Les valeurs admises de la propriété sont les suivantes :

**Valeur admise**

XMSC\_WPM\_TARGET\_TYPE\_BUSMEMBER

**Explication**

Le nom du groupe cible est le nom du membre de bus. Le groupe cible correspond à l'ensemble des moteurs de messagerie du membre de bus.

XMSC\_WPM\_TARGET\_TYPE\_CUSTOM

Le nom du groupe cible est le nom d'un groupe de moteurs de messagerie défini par l'utilisateur. Le groupe cible correspond à l'ensemble des moteurs de messagerie enregistrés avec le groupe défini par l'utilisateur.

XMSC\_WPM\_TARGET\_TYPE\_ME

Le nom du groupe cible est le nom d'un moteur de messagerie. Le groupe cible correspond au moteur de messagerie spécifié.

Par défaut, la propriété n'est pas définie.

***XMSC\_WPM\_TEMP\_Q\_PREFIX*****Type de données :**

String

**Propriété de :**

ConnectionFactory

Préfixe utilisé pour former le nom de la file d'attente temporaire créée dans le bus d'intégration de services lorsque l'application crée une file d'attente temporaire XMS. Le préfixe peut contenir jusqu'à 12 caractères.

Le nom de la file d'attente temporaire commence par les caractères "\_Q" suivis du préfixe. Le reste du nom est constitué de caractères générés par le système.

Par défaut, la propriété n'est pas définie, ce qui signifie que le nom d'une file d'attente temporaire n'a pas de préfixe.

Cette propriété est pertinente uniquement dans le domaine point à point.

***XMSC\_WPM\_TEMP\_TOPIC\_PREFIX*****Type de données :**

String

**Propriété de :**

ConnectionFactory

Préfixe utilisé pour former le nom d'une rubrique temporaire créée par l'application. Le préfixe peut contenir jusqu'à 12 caractères.

Le nom d'une file d'attente temporaire commence par les caractères "\_T" suivis du préfixe. Le reste du nom est constitué de caractères générés par le système.

Par défaut, la propriété n'est pas définie, ce qui signifie que le nom d'une rubrique temporaire n'a pas de préfixe.

Cette propriété est pertinente uniquement dans le domaine publication/abonnement.

### ***XMSC\_WPM\_TOPIC\_SPACE***

**Type de données :**

String

**Propriété de :**

Destination

**Nom utilisé dans un URI :**

topicSpace

Nom de l'espace de sujet contenant la rubrique. Seule une destination représentant une rubrique peut avoir cette propriété.

Par défaut, la propriété n'est pas définie, ce qui signifie que la rubrique par défaut est prise en charge.

Cette propriété est pertinente uniquement dans le domaine publication/abonnement.

## **Managed File Transfer Référence de développement de d'applications**

Informations de référence pour vous aider à développer des applications pour Managed File Transfer.

### **Exemples d'utilisation de `fteCreateTransfer` pour démarrer des programmes**

Vous pouvez utiliser la commande **`fteCreateTransfer`** pour spécifier les programmes à exécuter avant ou après un transfert.

Outre l'utilisation de **`fteCreateTransfer`**, il existe d'autres façons d'appeler un programme avant ou après un transfert. Pour plus d'informations, voir [Spécification des programmes à exécuter avec MFT](#).

Tous ces exemples utilisent la syntaxe suivante pour spécifier un programme:

```
[type:]commandspec[, [retrycount][, [retrywait][, successrc]]]
```

Pour plus d'informations sur cette syntaxe, voir [\*\*`fteCreateTransfer`\*\*: démarrer un nouveau transfert de fichiers](#).

#### **Exécution d'un programme exécutable**

L'exemple suivant spécifie un programme exécutable appelé `mycommand` et transmet deux arguments, `a` et `b`, au programme.

```
mycommand(a,b)
```

Pour exécuter ce programme sur l'agent source AGENT1 avant le démarrage du transfert, utilisez la commande suivante:

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -presrc mycommand(a,b)  
destinationSpecification sourceSpecification
```



## Exécution et relance d'un programme exécutable

L'exemple suivant spécifie un programme exécutable appelé `simple`, qui ne prend aucun argument. Une valeur de 1 est spécifiée pour `retrycount` et une valeur de 5 est spécifiée pour `retrywait`. Ces valeurs signifient que le programme sera retenté une fois s'il ne renvoie pas un code retour réussi, après une attente de cinq secondes. Aucune valeur n'étant spécifiée pour `successrc`, le seul code retour réussi est la valeur par défaut 0.

```
executable:simple,1,5
```

Pour exécuter ce programme sur l'agent source AGENT1 une fois le transfert terminé, utilisez la commande suivante:

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -postsrc executable:simple,1,5  
destinationSpecification sourceSpecification
```

## Exécution d'un script Ant et spécification de codes retour ayant abouti


L'exemple suivant spécifie un script Ant appelé `myscript` et transmet deux propriétés au script. Le script est exécuté à l'aide de la commande **fteAnt**. La valeur de `successrc` est définie sur `>2&<7&!5|0|14`, ce qui indique que les codes retour 0, 3, 4, 6 et 14 indiquent une réussite.

```
antscript:myscript(prop1=fred,prop2=bob),,,>2&<7&!5|0|14
```

Pour exécuter ce programme sur l'agent de destination AGENT2 avant le début du transfert, utilisez la commande suivante:

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -predst  
"antscript:myscript(prop1=fred,prop2=bob),,,>2&<7&!5|0|14"destinationSpecification sourceSpecification
```

## Exécution d'un script Ant et spécification de cibles à appeler

L'exemple suivant spécifie un script Ant appelé `script2` et deux cibles, `target1` et `target2`, à appeler. La propriété `prop1` est également transmise, avec la valeur `recmfm(F,B)`. Les parenthèses, les virgules (,) et les barres obliques inversées (\) sont des caractères spéciaux dans les commandes MFT et doivent être échappés avec une barre oblique inversée (\).  Les chemins d'accès aux fichiers sous Windows peuvent être spécifiés à l'aide de doubles barres obliques inversées (\\) comme séparateur ou à l'aide de barres obliques simples (/). Dans l'exemple suivant, la virgule (,) et les parenthèses sont échappés à l'aide d'une barre oblique inversée (\).

```
antscript:script2(target1,target2,prop1=recmfm\F\B\),,,>2&<7&!5|0|14
```

Pour exécuter ce programme sur l'agent de destination AGENT2 une fois le transfert terminé, utilisez la commande suivante:

```
fteCreateTransfer -sa AGENT1 -da AGENT2  
-postdst "antscript:script2(target1,target2,prop1=recmfm\F\B\),,,>2&<7&!5|0|14"  
destinationSpecification sourceSpecification
```

## Utilisation de métadonnées dans un script Ant

Vous pouvez spécifier une tâche Ant comme l'un des appels suivants pour un transfert:

- Pré-source
- Post-source
- prédestination
- post-destination

Lorsque la tâche Ant est exécutée, les métadonnées utilisateur du transfert sont rendues disponibles à l'aide de variables d'environnement. Vous pouvez accéder à ces données à l'aide, par exemple, du code suivant:

```
<property environment="environment" />
<echo>${environment.mymetadata}</echo>
```

où mymetadata est le nom de certaines métadonnées insérées dans le transfert.

### Exécution d'un script JCL

L'exemple suivant indique un script JCL appelé ZOSBATCH. Une valeur de 3 est spécifiée pour `retrycount`, une valeur de 30 est spécifiée pour `retrywait` et une valeur de 0 est spécifiée pour `successrc`. Ces valeurs signifient que le script est retenté trois fois s'il ne renvoie pas un code retour de réussite de 0, avec une attente de trente secondes entre chaque tentative.

```
jcl:ZOSBATCH,3,30,0
```

où ZOSBATCH est un membre d'un PDS appelé MYSYS.JCL et le fichier `agent.properties` contient la ligne `commandPath=...:/'MYSYS.JCL':...`

Pour exécuter ce programme sur l'agent source AGENT1 une fois le transfert terminé, utilisez la commande suivante:

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -postsrc jcl:ZOSBATCH,3,30,0
destinationSpecification sourceSpecification
```

### Tâches associées

[Spécification des programmes à exécuter avec MFT](#)

### Référence associée

[fteCreateTransfer](#): démarrer un nouveau transfert de fichier

## fteAnt: exécuter les tâches Ant dans MFT

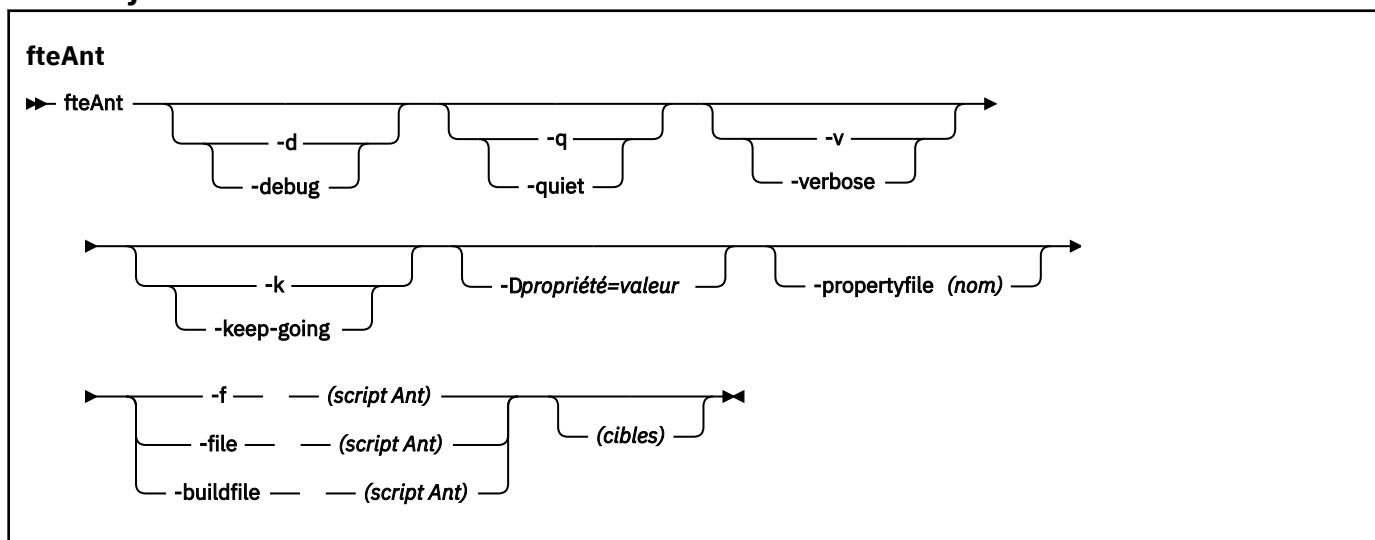
La commande **fteAnt** exécute des scripts Ant dans un environnement où des tâches Managed File Transfer Ant sont disponibles. Contrairement à la commande **ant** standard, **fteAnt** requiert que vous définissiez un fichier script.

### Tâches MFT Ant et paramètres imbriqués

Managed File Transfer fournit un certain nombre de tâches Ant que vous pouvez utiliser pour accéder aux fonctions de transfert de fichiers. Un ensemble de paramètres imbriqués est également disponible ; ces paramètres décrivent des ensembles imbriqués d'éléments communs à plusieurs des tâches Ant fournies.

La syntaxe de la commande **fteAnt**, les paramètres, l'exemple d'utilisation et les codes retour sont décrits dans le reste de cette rubrique. Pour plus de détails sur les tâches Ant et les paramètres imbriqués fournis par MFT, voir les sous-rubriques.

## Syntaxe de fteAnt



### Paramètres

#### **-debug ou -d**

Facultatif. Générez une sortie de débogage.

#### **-quiet ou -q**

Facultatif. Générer une sortie minimale.

#### **-verbose ou -v**

Facultatif. Générer une sortie prolixe.

#### **-continue ou -k**

Facultatif. Exécuter toutes les cibles qui ne dépendent pas des cibles ayant échoué.

#### **-D propriété=valeur**

Facultatif. Utilisez *value* pour une propriété *donnée*. Les propriétés définies avec **-D** sont prioritaires par rapport à celles définies dans un fichier de propriétés.

Utilisez la propriété **com.ibm.wmqfte.propertyset** pour spécifier l'ensemble des options de configuration utilisées pour les tâches Ant. Utilisez le nom d'un gestionnaire de files d'attente de coordination autre que celui par défaut comme valeur de cette propriété. Les tâches Ant utilisent ensuite l'ensemble des options de configuration associées à ce gestionnaire de files d'attente de coordination autre que celui par défaut. Si vous ne spécifiez pas cette propriété, l'ensemble d'options de configuration par défaut basé sur le gestionnaire de files d'attente de coordination par défaut est utilisé. Si vous spécifiez l'attribut **cmdqm** pour une tâche Ant, cet attribut est prioritaire sur l'ensemble des options de configuration spécifiées pour la commande **fteAnt**. Ce comportement s'applique que vous utilisiez l'ensemble d'options de configuration par défaut ou que vous spécifiez un ensemble avec la propriété **com.ibm.wmqfte.propertyset**.

#### **-propertyfile (nom)**

Facultatif. Chargez toutes les propriétés à partir d'un fichier dont les propriétés **-D** sont prioritaires.

#### **-f (script Ant), -file (script Ant) ou -buildfile (script Ant)**

Obligatoire. Indique le nom du script Ant à exécuter.

#### **cibles**

Facultatif. Nom d'une ou de plusieurs cibles à exécuter à partir du script Ant. Si vous ne spécifiez pas de valeur pour ce paramètre, la cible par défaut du script est exécutée.

#### **-version**

Facultatif. Affiche la commande Managed File Transfer et les versions de Ant.

#### **-? ou -h**

Facultatif. Affiche la syntaxe de la commande.

## Exemple

Dans cet exemple, la cible **copy** dans le script Ant `fte_script.xml` est exécutée et la commande écrit la sortie de débogage dans la sortie standard.

```
fteAnt -d -f fte_script.xml copy
```

## Codes retour

**0**

La commande a abouti.

**1**

La commande a échoué.

D'autres codes retour d'état peuvent également être spécifiés à partir de scripts Ant, par exemple à l'aide de la tâche d'échec Ant .

Pour plus d'informations, voir [Echec](#) .

## Concepts associés

[Initiation à l'utilisation des scripts Ant avec MFT](#)

## Tâches associées

[Utilisation de Apache Ant avec MFT](#)

## Référence associée

[Exemples de tâches Ant pour MFT](#)

## fte: awaitissue Ant tâche

Attend la fin d'une opération **fte:filecopy**, **fte:filemove** ou **fte:call** .

## Attributs

**id**

Obligatoire. Identifie le transfert à partir duquel attendre un résultat. En règle générale, il s'agit d'une propriété définie par l'attribut `idProperty` des tâches [fte:filecopy](#), [fte:filemove](#) ou [fte:call](#) .

**rcproperty**

Obligatoire. Nomme une propriété dans laquelle stocker le code retour de la tâche **fte:awaitoutcome** .

**dépassement du délai d'attente**

Facultatif. Durée maximale, en secondes, d'attente de la fin de l'opération. Le délai d'attente minimal est d'une seconde. Si vous ne spécifiez pas de valeur de délai d'attente, la tâche **fte:awaitoutcome** attend indéfiniment que le résultat de l'opération soit déterminé.

## Exemple

Dans cet exemple, une copie de fichier est démarrée et son identificateur est stocké dans la propriété `copy.id` . Pendant la progression de la copie, un autre traitement peut être effectué. L'instruction **fte:awaitoutcome** est utilisée pour attendre la fin de l'opération de copie. L'instruction **fte:awaitoutcome** identifie l'opération à attendre pour utiliser l'identificateur stocké dans la propriété `copy.id` . **fte:awaitoutcome** stocke un code retour indiquant le résultat de l'opération de copie dans une propriété appelée `copy.result` .

```
<-- issue a file copy request -->
<fte:filecopy
src="AGENT1@QM1"
dst="AGENT2@QM2"
idproperty="copy.id"
outcome="defer">

<fte:filespec
srcfilespec="/home/fteuser1/file.bin"
```

```
dstdir="/home/fteuser2"/>
</fte:filecopy>
<fte:awaitoutcome id="{copy.id}" rcProperty="copy.rc"/>
<echo>Copy id="{copy.id}" rc="{copy.rc}"</echo>
```

## Tâches associées

[Utilisation de Apache Ant avec MFT](#)

## fte: appel de la tâche Ant

Vous pouvez utiliser la tâche **fte:call** pour appeler à distance des scripts et des programmes.

Cette tâche vous permet d'envoyer une demande **fte:call** à un agent. L'agent traite cette demande en exécutant un script ou un programme et en renvoyant le résultat. Les commandes à appeler doivent être accessibles à l'agent. Vérifiez que la valeur de la propriété `commandPath` dans le fichier `agent.properties` inclut l'emplacement des commandes à appeler. Les informations de chemin spécifiées par l'élément imbriqué de commande doivent être relatives aux emplacements spécifiés par la propriété `commandPath`. Par défaut, `commandPath` est vide, de sorte que l'agent ne peut pas appeler de commandes. Pour plus d'informations sur cette propriété, voir [commandPath MFT property](#).

Pour plus d'informations sur le fichier `agent.properties`, voir [Le fichier MFT agent.properties](#).

## Attributs

### agent

Obligatoire. Indique l'agent sur lequel la demande **fte:call** doit être soumise. Spécifiez les informations d'agent sous la forme suivante: `agentname@qmgrname`, où `agentname` est le nom de l'agent et `qmgrname` est le nom du gestionnaire de files d'attente auquel cet agent est directement connecté.

### cmdqm

Facultatif. Gestionnaire de files d'attente de commandes auquel soumettre la demande. Spécifiez ces informations sous la forme `qmgrname@host@port@channel`, où:

- `qmgrname` est le nom du gestionnaire de files d'attente
- `host` est le nom d'hôte facultatif du système sur lequel le gestionnaire de files d'attente est en cours d'exécution.
- `port` est le numéro de port facultatif sur lequel le gestionnaire de files d'attente est en mode écoute
- `channel` est le canal SVRCONN facultatif à utiliser

Si vous omettez les informations `host`, `port` ou `channel` pour le gestionnaire de files d'attente de commandes, les informations de connexion spécifiées dans le fichier `command.properties` sont utilisées.



**Avertissement :** Si aucune valeur n'est spécifiée pour:

- Variable `host`, le mode de liaison est utilisé
- Variable `port`, la valeur 1414 est utilisée
- Variable `channel`, `SYSTEM.DEF.SVRCONN` est utilisé.

Voir [Le fichier MFT command.properties](#) pour plus d'informations.

Toutefois, vous ne pouvez pas ignorer les attributs du milieu, par exemple, `qmgrname@host@@channel`. Vous pouvez, par exemple, avoir `qmgrname@host`, ou `qmgrname@host@port`, ou `qmgrname@hostport@@channel`.

MFT fractionne l'attribut donné à l'aide du délimiteur `@`. En fonction du nombre de jetons trouvés, il prend le premier jeton comme `qmgrname`, le second comme `host`, le troisième comme `port` et enfin le canal.

Pour plus d'informations, voir [Le fichier MFT command.properties](#).

Vous pouvez utiliser la propriété **com.ibm.wmqfte.propertySet** pour spécifier le fichier `command.properties` à utiliser. Pour plus d'informations, voir [com.ibm.wmqfte.propertySet](#).

Si vous n'utilisez pas l'attribut `cmdqm`, la tâche utilise par défaut la propriété `com.ibm.wmqfte.ant.commandQueueManager`, si cette propriété est définie. Si la propriété `com.ibm.wmqfte.ant.commandQueueManager` n'est pas définie, une tentative de connexion au gestionnaire de files d'attente par défaut, défini dans le fichier `command.properties`, est effectuée. Le format de la propriété `com.ibm.wmqfte.ant.commandQueueManager` est identique à l'attribut `cmdqm`, c'est-à-dire `qmgrname@host@port@channel`.

### idproperty

Facultatif sauf si vous avez spécifié un `outcome` de `defer`. Indique le nom d'une propriété à laquelle affecter l'identificateur de transfert. Les identificateurs de transfert sont générés au moment où une demande de transfert est soumise et vous pouvez utiliser des identificateurs de transfert pour suivre la progression d'un transfert, diagnostiquer les problèmes liés à un transfert et annuler un transfert.

Vous ne pouvez pas spécifier cette propriété si vous avez également spécifié une propriété `outcome` de `ignore`. Toutefois, vous devez spécifier `idproperty` si vous avez également spécifié une propriété `outcome` de `defer`.

### Nom de travail

Facultatif. Affecte un nom de travail à la demande **fte:call**. Vous pouvez utiliser des noms de travail pour créer des groupes logiques de transferts. Utilisez la tâche «tâche fte: uuid Ant», à la page [2210](#) pour générer des noms de travail pseudo-uniqes. Si vous n'utilisez pas l'attribut `jobname`, la tâche utilise par défaut la valeur de la propriété `com.ibm.wmqfte.ant.jobName`, si cette propriété est définie. Si vous ne définissez pas cette propriété, aucun nom de travail n'est associé à la demande **fte:call**.

### utilisateur d'origine

Facultatif. Indique l'ID utilisateur d'origine à associer à la demande **fte:call**. Si vous n'utilisez pas l'attribut `origuser`, la tâche utilise par défaut l'ID utilisateur utilisé pour exécuter le script Ant.

### résultat

Facultatif. Détermine si la tâche attend la fin de l'opération **fte:call** avant de renvoyer le contrôle au script Ant. Indiquez l'une des options suivantes :

#### attendre

La tâche attend la fin de l'opération **fte:call** avant de revenir. Lorsqu'un `outcome` de `await` est spécifié, l'attribut `idproperty` est facultatif.

#### reporter

La tâche est renvoyée dès que la demande **fte:call** a été soumise et suppose que le résultat de l'opération d'appel est traité ultérieurement à l'aide des tâches `awaitresultant` ou `ignoreissue`. Lorsqu'une valeur `outcome` de `defer` est spécifiée, l'attribut `idproperty` est obligatoire.

#### ignorer

Si le résultat de l'opération **fte:call** n'est pas important, vous pouvez spécifier la valeur `ignore`. La tâche est ensuite renvoyée dès que la demande **fte:call** a été soumise, sans allouer de ressources pour le suivi du résultat de la commande. Lorsqu'une valeur `outcome` de `ignore` est spécifiée, l'attribut `idproperty` ne peut pas être spécifié.

Si vous ne spécifiez pas l'attribut `issue`, la tâche utilise par défaut la valeur `await`.

### rcproperty

Facultatif. Indique le nom d'une propriété à laquelle affecter le code de résultat de la demande **fte:call**. Le code de résultat reflète le résultat global de la demande **fte:call**.

Vous ne pouvez pas spécifier cette propriété si vous avez également spécifié une propriété `outcome` `ignore` ou `defer`. Toutefois, vous devez spécifier `rcproperty` si vous avez spécifié un résultat de `await`.

## Paramètres spécifiés en tant qu'éléments imbriqués

### **fte: commande**

Indique la commande à appeler par l'agent. Vous ne pouvez associer qu'un seul élément **fte: command** à une opération **fte: call** donnée. La commande à appeler doit se trouver dans le chemin spécifié par la propriété `commandPath` dans le fichier `agent.properties` de l'agent.

### **fte: métadonnées**

Vous pouvez spécifier des métadonnées à associer à l'opération d'appel. Ces métadonnées sont enregistrées dans les messages de journal générés par l'opération d'appel. Vous ne pouvez associer qu'un seul bloc de métadonnées à un élément de transfert donné ; cependant, ce bloc peut contenir de nombreux éléments de métadonnées.

### Exemple

Cet exemple montre comment appeler une commande à l'adresse AGENT1 s'exécutant sur le gestionnaire de files d'attente QM1. La commande à appeler est le script `command.sh`, et le script est appelé avec un seul argument `xyz`. La commande `command.sh` se trouve dans le chemin spécifié par la propriété `commandPath` dans le fichier `agent.properties` de l'agent.

```
<fte:call cmdqm="QM0@localhost@1414@SYSTEM.DEF.SVRCONN"
  agent="AGENT1@QM1"
  rcproperty="call.rc"
  origuser="bob"
  jobname="{job.id}">
  <fte:command command="command.sh" successrc="1" retrycount="5" retrywait="30">
    <fte:arg value="xyz"/>
  </fte:command>
  <fte:metadata>
    <fte:entry name="org.foo.accountName" value="BDG3R"/>
  </fte:metadata>
</fte:call>
```

### Tâches associées

[Utilisation de Apache Ant avec MFT](#)

## **fte: annuler la tâche Ant**

Annule un transfert géré ou un appel géré Managed File Transfer . Un transfert géré peut avoir été créé à l'aide des tâches **fte:filecopy** ou **fte:filemove** . Un appel géré a peut-être été créé à l'aide de la tâche **fte:call** .

### Attributs

#### **agent**

Obligatoire. Indique l'agent sur lequel la demande **fte:cancel** doit être soumise. La valeur se présente sous la forme suivante: `agentname@qmgrname` , où `agentname` est le nom de l'agent et `qmgrname` est le nom du gestionnaire de files d'attente auquel cet agent est directement connecté.

#### **cmdqm**

Facultatif. Gestionnaire de files d'attente de commandes auquel soumettre la demande. Spécifiez ces informations sous la forme `qmgrname@host@port@channel`, où:

- `qmgrname` est le nom du gestionnaire de files d'attente
- `host` est le nom d'hôte facultatif du système sur lequel le gestionnaire de files d'attente est en cours d'exécution.
- `port` est le numéro de port facultatif sur lequel le gestionnaire de files d'attente est en mode écoute
- `channel` est le canal SVRCONN facultatif à utiliser

Si vous omettez les informations *host*, *port* ou *channel* pour le gestionnaire de files d'attente de commandes, les informations de connexion spécifiées dans le fichier `command.properties` sont utilisées.



**Avertissement** : Si aucune valeur n'est spécifiée pour:

- Variable *host* , le mode de liaison est utilisé
- Variable *port* , la valeur 1414 est utilisée
- Variable *channel* , SYSTEM.DEF.SVRCONN est utilisé.

Voir [Le fichier MFT command.properties](#) pour plus d'informations.

Toutefois, vous ne pouvez pas ignorer les attributs du milieu, par exemple, `qmgrname@host@@channel`. Vous pouvez, par exemple, avoir `qmgrname@host`, ou `qmgrname@host@port`, ou `qmgrname@hostport@@channel`.

MFT fractionne l'attribut donné à l'aide du délimiteur @ . En fonction du nombre de jetons trouvés, il prend le premier jeton comme *qmgrname*, le second comme *host*, le troisième comme *port* et enfin le *canal*.

Pour plus d'informations, voir [Le fichier MFT command.properties](#).

Vous pouvez utiliser la propriété **com.ibm.wmqfte.propertySet** pour spécifier le fichier `command.properties` à utiliser. Pour plus d'informations, voir [com.ibm.wmqfte.propertySet](#).

Si vous n'utilisez pas l'attribut `cmdqm` , la tâche utilise par défaut la propriété `com.ibm.wmqfte.ant.commandQueueManager` , si cette propriété est définie. Si la propriété `com.ibm.wmqfte.ant.commandQueueManager` n'est pas définie, une tentative de connexion au gestionnaire de files d'attente par défaut, défini dans le fichier `command.properties` , est effectuée. Le format de la propriété `com.ibm.wmqfte.ant.commandQueueManager` est identique à l'attribut `cmdqm` , c'est-à-dire `qmgrname@host@port@channel`.

## id

Obligatoire. Indique l'identificateur de transfert du transfert à annuler. Les identificateurs de transfert sont générés au moment où une demande de transfert est soumise par les tâches [fte:filecopy](#) et [fte:filemove](#) .

## utilisateur d'origine

Facultatif. Indique l'ID utilisateur d'origine à associer à la demande **cancel** . Si l'attribut `origuser` n'est pas utilisé, la tâche utilise par défaut l'ID utilisateur utilisé pour exécuter le script Ant .

## Exemple

L'exemple envoie une demande **fte:cancel** au gestionnaire de files d'attente de commandes `qm0`. La demande **fte:cancel** est ciblée sur `agent1` sur le gestionnaire de files d'attente `qm1` pour l'identificateur de transfert renseigné par la variable `transfer.id` . La demande est exécutée à l'aide de l'ID utilisateur "bob" .

```
<fte:cancel cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
  agent="agent1@qm1"
  id="${transfer.id}"
  origuser="bob"/>
```

## Tâches associées

[Utilisation de Apache Ant avec MFT](#)

## tâche fte:filecopy Ant

La tâche **fte:filecopy** copie les fichiers entre les agents Managed File Transfer . Le fichier n'est pas supprimé de l'agent source.



## Attributs

### cmdqm

Facultatif. Gestionnaire de files d'attente de commandes auquel soumettre la demande. Spécifiez ces informations sous la forme *qmgrname@host@port@channel*, où:

- *qmgrname* est le nom du gestionnaire de files d'attente
- *host* est le nom d'hôte facultatif du système sur lequel le gestionnaire de files d'attente est en cours d'exécution.
- *port* est le numéro de port facultatif sur lequel le gestionnaire de files d'attente est en mode écoute
- *channel* est le canal SVRCONN facultatif à utiliser

Si vous omettez les informations *host*, *port* ou *channel* pour le gestionnaire de files d'attente de commandes, les informations de connexion spécifiées dans le fichier `command.properties` sont utilisées.



**Avertissement** : Si aucune valeur n'est spécifiée pour:

- Variable *host* , le mode de liaison est utilisé
- Variable *port* , la valeur 1414 est utilisée
- Variable *channel* , SYSTEM.DEF.SVRCONN est utilisé.

Voir [Le fichier MFT command.properties](#) pour plus d'informations.

Toutefois, vous ne pouvez pas ignorer les attributs du milieu, par exemple, *qmgrname@host@@channel*. Vous pouvez, par exemple, avoir *qmgrname@host*, ou *qmgrname@host@port*, ou *qmgrname@hostport@@channel*.

MFT fractionne l'attribut donné à l'aide du délimiteur @ . En fonction du nombre de jetons trouvés, il prend le premier jeton comme *qmgrname*, le second comme *host*, le troisième comme *port* et enfin le *canal*.

Pour plus d'informations, voir [Le fichier MFT command.properties](#).

Vous pouvez utiliser la propriété **com.ibm.wmqfte.propertySet** pour spécifier le fichier `command.properties` à utiliser. Pour plus d'informations, voir [com.ibm.wmqfte.propertySet](#).

Si vous n'utilisez pas l'attribut `cmdqm` , la tâche utilise par défaut la propriété `com.ibm.wmqfte.ant.commandQueueManager` , si cette propriété est définie. Si la propriété `com.ibm.wmqfte.ant.commandQueueManager` n'est pas définie, une tentative de connexion au gestionnaire de files d'attente par défaut, défini dans le fichier `command.properties` , est effectuée. Le format de la propriété `com.ibm.wmqfte.ant.commandQueueManager` est identique à l'attribut `cmdqm` , c'est-à-dire *qmgrname@host@port@channel*.

### Outils de maintenance en mode dédié

Obligatoire. Indique l'agent de destination pour l'opération de copie. Spécifiez ces informations sous la forme suivante: *agentname@qmgrname* où *agentname* est le nom de l'agent de destination et *qmgrname* est le nom du gestionnaire de files d'attente auquel cet agent est directement connecté.

### idproperty

Facultatif sauf si vous avez spécifié un `outcome` de `defer`. Indique le nom d'une propriété à laquelle affecter l'identificateur de transfert. Les identificateurs de transfert sont générés au moment où une demande de transfert est soumise et vous pouvez utiliser des identificateurs de transfert pour suivre la progression d'un transfert, diagnostiquer les problèmes liés à un transfert et annuler un transfert.

Vous ne pouvez pas spécifier cette propriété si vous avez également spécifié une propriété `outcome` de `ignore`. Toutefois, vous devez spécifier `idproperty` si vous avez également spécifié une propriété `outcome` de `defer`.

### Nom de travail

Facultatif. Affecte un nom de travail à la demande de copie. Vous pouvez utiliser des noms de travail pour créer des groupes logiques de transferts. Utilisez la tâche «[tâche fte: uuid Ant](#)», à la page 2210 pour générer des noms de travail pseudo-uniqes. Si vous n'utilisez pas l'attribut `jobname` , la tâche

utilise par défaut la valeur de la propriété `com.ibm.wmqfte.ant.jobName`, si cette propriété est définie. Si vous ne définissez pas cette propriété, aucun nom de travail n'est associé à la demande de copie.

### utilisateur d'origine

Facultatif. Indique l'ID utilisateur d'origine à associer à la demande de copie. Si vous n'utilisez pas l'attribut `origuser`, la tâche utilise par défaut l'ID utilisateur utilisé pour exécuter le script Ant.

### résultat

Facultatif. Détermine si la tâche attend la fin de l'opération de copie avant de renvoyer le contrôle au script Ant. Indiquez l'une des options suivantes :

#### attendre

La tâche attend la fin de l'opération de copie avant de revenir. Lorsqu'un outcome de `await` est spécifié, l'attribut `idproperty` est facultatif.

#### reporter

La tâche est renvoyée dès que la demande de copie a été soumise et suppose que le résultat de l'opération de copie est traité ultérieurement à l'aide des tâches «`fte: awaitissue Ant tâche`», à la page 2196 ou «`tâche fte: ignoreissue Ant`», à la page 2208. Lorsqu'une valeur outcome de `defer` est spécifiée, l'attribut `idproperty` est obligatoire.

#### ignorer

Si le résultat de l'opération de copie n'est pas important, vous pouvez spécifier la valeur `ignore`. La tâche revient ensuite dès que la demande de copie a été soumise, sans allouer de ressources pour suivre le résultat du transfert. Lorsqu'une valeur outcome de `ignore` est spécifiée, l'attribut `idproperty` ne peut pas être spécifié.

Si vous ne spécifiez pas l'attribut `issue`, la tâche utilise par défaut la valeur `await`.

### priority

Facultatif. Indique la priorité à associer à la demande de copie. En général, les demandes de transfert de priorité plus élevée sont prioritaires par rapport aux demandes de priorité plus faible. La valeur de priorité doit être comprise entre 0 et 9 (inclus). Une valeur de priorité de 0 correspond à la priorité la plus faible et une valeur de 9 à la priorité la plus élevée. Si vous ne spécifiez pas l'attribut `priority`, le transfert prend par défaut la valeur 0.

### rcproperty

Facultatif. Indique le nom d'une propriété à laquelle affecter le code de résultat de la demande de copie. Le code de résultat reflète le résultat global de la demande de copie.

Vous ne pouvez pas spécifier cette propriété si vous avez également spécifié une propriété outcome `ignore` ou `defer`. Toutefois, vous devez spécifier `rcproperty` si vous spécifiez un résultat `await`.

### transferRecoveryTimeout

Facultatif. Définit la durée, en secondes, pendant laquelle un agent source continue de tenter de récupérer un transfert de fichiers bloqué. Indiquez l'une des options suivantes :

#### -1

L'agent continue de tenter de reprendre le transfert bloqué jusqu'à ce que ce dernier soit terminé. Cette option est équivalente au comportement par défaut de l'agent si la propriété n'est pas définie.

#### 0

L'agent arrête le transfert de fichiers dès qu'il entre dans la phase de reprise.

#### >0

L'agent continue de tenter de reprendre le transfert bloqué pendant le nombre de secondes défini par l'entier positif spécifié. Exemple :

```
<fte:filecopy cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
  src="agent1@qm1" dst="agent2@qm2"
  rcproperty="copy.result" transferRecoveryTimeout="21600">
  <fte:filespec srcfilespec="/home/fteuser1/file.bin" dstfile="/home/fteuser2/
file.bin"/>
</fte:filecopy>
```

indique que l'agent tente de récupérer le transfert pendant 6 heures à partir du moment où il entre en récupération. La valeur maximale de cet attribut est 999999999.

La spécification de la valeur de délai d'attente de reprise de transfert de cette manière la définit par transfert. Pour définir une valeur globale pour tous les transferts dans un réseau Managed File Transfer, vous pouvez ajouter une propriété aux propriétés de délai de récupération de transfert. Pour plus d'informations, voir Timeout option for transfers in recovery.

#### **srce**

Obligatoire. Indique l'agent source pour l'opération de copie. Spécifiez ces informations sous la forme suivante: *agentname@qmgrname*, où *agentname* est le nom de l'agent source et *qmgrname* est le nom du gestionnaire de files d'attente auquel cet agent est directement connecté.

## **Paramètres spécifiés en tant qu'éléments imbriqués**

#### **fte: spéc\_fichier**

Obligatoire. Vous devez spécifier au moins une spécification de fichier qui identifie les fichiers à copier. Vous pouvez spécifier plusieurs spécifications de fichier si nécessaire. Pour plus d'informations, voir «fte: filespec Ant élément imbriqué», à la page 2210.

#### **fte: métadonnées**

Vous pouvez spécifier des métadonnées à associer à l'opération de copie. Ces métadonnées sont transportées avec le transfert et sont enregistrées dans les messages de journal générés par le transfert. Vous ne pouvez associer qu'un seul bloc de métadonnées à un élément de transfert donné ; cependant, ce bloc peut contenir de nombreux éléments de métadonnées. Pour plus d'informations, voir la rubrique fte: metadata.

#### **fte: presrc**

Indique un appel de programme à effectuer au niveau de l'agent source avant le début du transfert. Vous ne pouvez associer qu'un seul élément `fte: presrc` à un transfert donné. Pour plus d'informations, voir la rubrique Appel de programme.

#### **fte: predst**

Indique un appel de programme à effectuer au niveau de l'agent de destination avant le début du transfert. Vous ne pouvez associer qu'un seul élément `fte: predst` à un transfert donné. Pour plus d'informations, voir la rubrique Appel de programme.

#### **fte: postsrc**

Indique un appel de programme à effectuer au niveau de l'agent source une fois le transfert terminé. Vous ne pouvez associer qu'un seul élément `fte: postsrc` à un transfert donné. Pour plus d'informations, voir la rubrique Appel de programme.

#### **fte: postdst**

Indique un appel de programme à effectuer au niveau de l'agent de destination une fois le transfert terminé. Vous ne pouvez associer qu'un seul élément `fte: postdst` à un transfert donné. Pour plus d'informations, voir la rubrique Appel de programme.

Si `fte:presrc`, `fte:predst`, `fte:postsrc`, `fte:postdst` et que les exits ne renvoient pas de statut de réussite, les règles sont les suivantes dans l'ordre indiqué:

1. Exécutez les exits de démarrage de la source. Si les exits de démarrage de la source échouent, le transfert échoue et aucune autre opération n'est effectuée.
2. Exécutez l'appel présource (s'il est présent). Si l'appel de présource échoue, le transfert échoue et rien n'est exécuté.
3. Exécutez les exits de démarrage de destination. Si les exits de démarrage de la destination échouent, le transfert échoue et aucune autre opération n'est effectuée.
4. Exécutez l'appel de prédestination (s'il est présent). Si l'appel de prédestination échoue, le transfert échoue et rien n'est exécuté.
5. Effectuez les transferts de fichiers.
6. Exécutez les exits de fin de destination. Il n'y a pas de statut d'échec pour ces exits.

7. Si le transfert aboutit (si le transfert de certains fichiers aboutit, il est considéré comme ayant abouti), exécutez l'appel post-destination (s'il est présent). Si l'appel post-destination échoue, le transfert échoue.
8. Exécutez les exits d'extrémité source. Il n'y a pas de statut d'échec pour ces exits.
9. Si le transfert aboutit, exécutez l'appel post-source (s'il est présent). Si l'appel post-source échoue, le transfert échoue.

## Exemples

Cet exemple illustre un transfert de fichier de base entre agent1 et agent2. La commande permettant de démarrer le transfert de fichiers est envoyée à un gestionnaire de files d'attente appelé qm0, à l'aide d'une connexion en mode transport client. Le résultat de l'opération de transfert de fichier est affecté à la propriété appelée copy.result.

```
<fte:filecopy cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
  src="agent1@qm1" dst="agent2@qm2"
  rcproperty="copy.result">
  <fte:filespec srcfilespec="/home/fteuser1/file.bin" dstfile="/home/fteuser2/file.bin"/>
</fte:filecopy>
```

Cet exemple montre le même transfert de fichiers, mais avec l'ajout de métadonnées et le démarrage d'un programme au niveau de l'agent source une fois le transfert terminé.

```
<fte:filecopy cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
  src="agent1@qm1" dst="agent2@qm2"
  rcproperty="copy.result">
  <fte:metadata>
    <fte:entry name="org.example.departId" value="ACCOUNTS"/>
    <fte:entry name="org.example.batchGroup" value="A1"/>
  </fte:metadata>
  <fte:filespec srcfilespec="/home/fteuser1/file.bin" dstfile="/home/fteuser2/file.bin"/>
  <fte:postsrc command="/home/fteuser2/scripts/post.sh" successsrc="1" >
    <fte:arg value="/home/fteuser2/file.bin"/>
  </fte:postsrc>
</fte:filecopy>
```

## Concepts associés

[Option de délai d'attente pour les transferts de fichiers en cours de récupération](#)

## Tâches associées

[Utilisation de Apache Ant avec MFT](#)

## fte: tâche Ant filemove

La tâche **fte:filemove** déplace les fichiers entre les agents Managed File Transfer. Lorsqu'un fichier a été transféré avec succès de l'agent source vers l'agent cible, le fichier est supprimé de l'agent source.

## Attributs

### cmdqm

Facultatif. Gestionnaire de files d'attente de commandes auquel soumettre la demande. Spécifiez ces informations sous la forme *qmgrname@host@port@channel*, où:

- *qmgrname* est le nom du gestionnaire de files d'attente
- *host* est le nom d'hôte facultatif du système sur lequel le gestionnaire de files d'attente est en cours d'exécution.

- *port* est le numéro de port facultatif sur lequel le gestionnaire de files d'attente est en mode écoute
- *channel* est le canal SVRCONN facultatif à utiliser

Si vous omettez les informations *host*, *port* ou *channel* pour le gestionnaire de files d'attente de commandes, les informations de connexion spécifiées dans le fichier `command.properties` sont utilisées.



**Avertissement :** Si aucune valeur n'est spécifiée pour:

- Variable *host*, le mode de liaison est utilisé
- Variable *port*, la valeur 1414 est utilisée
- Variable *channel*, SYSTEM.DEF.SVRCONN est utilisé.

Voir [Le fichier MFT `command.properties`](#) pour plus d'informations.

Toutefois, vous ne pouvez pas ignorer les attributs du milieu, par exemple, `qmgrname@host@@channel`. Vous pouvez, par exemple, avoir `qmgrname@host`, ou `qmgrname@host@port`, ou `qmgrname@hostport@@channel`.

MFT fractionne l'attribut donné à l'aide du délimiteur @. En fonction du nombre de jetons trouvés, il prend le premier jeton comme *qmgrname*, le second comme *host*, le troisième comme *port* et enfin le *canal*.

Pour plus d'informations, voir [Le fichier MFT `command.properties`](#).

Vous pouvez utiliser la propriété **`com.ibm.wmqfte.propertySet`** pour spécifier le fichier `command.properties` à utiliser. Pour plus d'informations, voir [com.ibm.wmqfte.propertySet](#).

Si vous n'utilisez pas l'attribut `cmdqm`, la tâche utilise par défaut la propriété `com.ibm.wmqfte.ant.commandQueueManager`, si cette propriété est définie. Si la propriété `com.ibm.wmqfte.ant.commandQueueManager` n'est pas définie, une tentative de connexion au gestionnaire de files d'attente par défaut, défini dans le fichier `command.properties`, est effectuée. Le format de la propriété `com.ibm.wmqfte.ant.commandQueueManager` est identique à l'attribut `cmdqm`, c'est-à-dire `qmgrname@host@port@channel`.

### Outils de maintenance en mode dédié

Obligatoire. Indique l'agent de destination pour l'opération de copie. Spécifiez ces informations sous la forme suivante: `agentname@qmgrname` où *agentname* est le nom de l'agent de destination et *qmgrname* est le nom du gestionnaire de files d'attente auquel cet agent est directement connecté.

### idproperty

Facultatif sauf si vous avez spécifié un `outcome` de `defer`. Indique le nom d'une propriété à laquelle affecter l'identificateur de transfert. Les identificateurs de transfert sont générés au moment où une demande de transfert est soumise et vous pouvez utiliser des identificateurs de transfert pour suivre la progression d'un transfert, diagnostiquer les problèmes liés à un transfert et annuler un transfert.

Vous ne pouvez pas spécifier cette propriété si vous avez également spécifié une propriété `outcome` de `ignore`. Toutefois, vous devez spécifier `idproperty` si vous avez également spécifié une propriété `outcome` de `defer`.

### Nom de travail

Facultatif. Affecte un nom de travail à la demande de déplacement. Vous pouvez utiliser des noms de travail pour créer des groupes logiques de transferts. Utilisez la tâche `fte: uuid` pour générer des noms de travail pseudo-uniqes. Si vous n'utilisez pas l'attribut `jobname`, la tâche utilise par défaut la valeur de la propriété `com.ibm.wmqfte.ant.jobName`, si cette propriété est définie. Si vous ne définissez pas cette propriété, aucun nom de travail n'est associé à la demande de déplacement.

### utilisateur d'origine

Facultatif. Indique l>ID utilisateur d'origine à associer à la demande de déplacement. Si vous n'utilisez pas l'attribut `origuser`, la tâche utilise par défaut l>ID utilisateur utilisé pour exécuter le script `Ant`.

### résultat

Facultatif. Détermine si la tâche attend la fin de l'opération de déplacement avant de renvoyer le contrôle au script `Ant`. Indiquez l'une des options suivantes :

### attendre

La tâche attend la fin de l'opération de déplacement avant de revenir. Lorsqu'un outcome de await est spécifié, l'attribut idproperty est facultatif.

### reporter

La tâche est renvoyée dès que la demande de déplacement a été soumise et suppose que le résultat de l'opération de déplacement est traité ultérieurement à l'aide de la tâche «fte: awaitissue Ant tâche», à la page 2196 ou «tâche fte: ignoreissue Ant», à la page 2208 . Lorsqu'une valeur outcome de defer est spécifiée, l'attribut idproperty est obligatoire.

### ignorer

Si le résultat de l'opération de déplacement n'est pas important, vous pouvez spécifier la valeur ignore. La tâche revient ensuite dès que la demande de déplacement a été soumise, sans allouer de ressources pour le suivi du résultat du transfert. Lorsqu'une valeur outcome de ignore est spécifiée, l'attribut idproperty ne peut pas être spécifié.

Si vous ne spécifiez pas l'attribut issue , la tâche utilise par défaut la valeur await.

### priority

Facultatif. Indique la priorité à associer à la demande de déplacement. En général, les demandes de transfert de priorité plus élevée sont prioritaires par rapport aux demandes de priorité plus faible. La valeur de priorité doit être comprise entre 0 et 9 (inclus). Une valeur de priorité de 0 correspond à la priorité la plus faible et une valeur de 9 à la priorité la plus élevée. Si vous ne spécifiez pas l'attribut priority , le transfert prend par défaut la valeur 0.

### rcproperty

Facultatif. Indique le nom d'une propriété à laquelle affecter le code de résultat de la demande de déplacement. Le code de résultat reflète le résultat global de la demande de déplacement.

Vous ne pouvez pas spécifier cette propriété si vous avez également spécifié une propriété outcome ignore ou defer. Toutefois, vous devez spécifier rcproperty si vous avez spécifié un résultat de await.

### transferRecoveryTimeout

Facultatif. Définit la durée, en secondes, pendant laquelle un agent source continue de tenter de récupérer un transfert de fichiers bloqué. Indiquez l'une des options suivantes :

#### -1

L'agent continue de tenter de reprendre le transfert bloqué jusqu'à ce que ce dernier soit terminé. Cette option est équivalente au comportement par défaut de l'agent si la propriété n'est pas définie.

#### 0

L'agent arrête le transfert de fichiers dès qu'il entre dans la phase de reprise.

#### >0

L'agent continue de tenter de reprendre le transfert bloqué pendant le nombre de secondes défini par l'entier positif spécifié. Exemple :

```
<fte:filemove cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
  src=agent1@qm1 dst="agent2@qm2"
  rcproperty="move.result" transferRecoveryTimeout="21600">
  <fte:filespec srcfilespec="/home/fteuser1/file.bin" dstfile="/home/fteuser2/
file.bin"/>
</fte:filemove
```

indique que l'agent tente de récupérer le transfert pendant 6 heures à partir du moment où il entre en récupération. La valeur maximale de cet attribut est 999999999.

La spécification de la valeur de délai d'attente de reprise de transfert de cette manière la définit par transfert. Pour définir une valeur globale pour tous les transferts dans un réseau Managed File Transfer , vous pouvez ajouter une propriété aux propriétés de délai de récupération de transfert. Pour plus d'informations, voir Timeout option for transfers in recovery.

**srce**

Obligatoire. Indique l'agent source pour l'opération de déplacement. Spécifiez ces informations sous la forme suivante: *agentname@qmgrname* où *agentname* est le nom de l'agent source et *qmgrname* est le nom du gestionnaire de files d'attente auquel cet agent est directement connecté.

**Paramètres spécifiés en tant qu'éléments imbriqués****fte: spéc\_fichier**

Obligatoire. Vous devez spécifier au moins une spécification de fichier qui identifie les fichiers à déplacer. Vous pouvez spécifier plusieurs spécifications de fichier si nécessaire. Pour plus d'informations, voir «[fte: filespec Ant élément imbriqué](#)», à la page 2210.

**fte: métadonnées**

Facultatif. Vous pouvez spécifier des métadonnées à associer à l'opération de déplacement de fichier. Ces métadonnées sont transportées avec le transfert et sont enregistrées dans les messages de journal générés par le transfert. Vous ne pouvez associer qu'un seul bloc de métadonnées à un élément de transfert donné ; cependant, ce bloc peut contenir de nombreux éléments de métadonnées. Pour plus d'informations, voir la rubrique [fte: metadata](#) .

**fte: presrc**

Facultatif. Indique un appel de programme à effectuer au niveau de l'agent source avant le début du transfert. Vous ne pouvez associer qu'un seul élément `fte: presrc` à un transfert donné. Pour plus d'informations, voir la rubrique [Appel de programme](#) .

**fte: predst**

Facultatif. Indique un appel de programme à effectuer au niveau de l'agent de destination avant le début du transfert. Vous ne pouvez associer qu'un seul élément `fte: predst` à un transfert donné. Pour plus d'informations, voir la rubrique [Appel de programme](#) .

**fte: postsrc**

Facultatif. Indique un appel de programme à effectuer au niveau de l'agent source une fois le transfert terminé. Vous ne pouvez associer qu'un seul élément `fte: postsrc` à un transfert donné. Pour plus d'informations, voir la rubrique [Appel de programme](#) .

**fte: postdst**

Facultatif. Indique un appel de programme à effectuer au niveau de l'agent de destination une fois le transfert terminé. Vous ne pouvez associer qu'un seul élément `fte: postdst` à un transfert donné. Pour plus d'informations, voir la rubrique [Appel de programme](#) .

Si `fte:presrc`, `fte:predst`, `fte:postsrc`, `fte:postdst` et que les exits ne renvoient pas de statut de réussite, les règles sont les suivantes dans l'ordre indiqué:

1. Exécutez les exits de démarrage de la source. Si les exits de démarrage de la source échouent, le transfert échoue et aucune autre opération n'est effectuée.
2. Exécutez l'appel présorce (s'il est présent). Si l'appel de présorce échoue, le transfert échoue et rien n'est exécuté.
3. Exécutez les exits de démarrage de destination. Si les exits de démarrage de la destination échouent, le transfert échoue et aucune autre opération n'est effectuée.
4. Exécutez l'appel de prédestination (s'il est présent). Si l'appel de prédestination échoue, le transfert échoue et rien n'est exécuté.
5. Effectuez les transferts de fichiers.
6. Exécutez les exits de fin de destination. Il n'y a pas de statut d'échec pour ces exits.
7. Si le transfert aboutit (si certains fichiers sont transférés avec succès, le transfert est considéré comme ayant abouti), exécutez l'appel post-destination (s'il est présent). Si l'appel post-destination échoue, le transfert échoue.
8. Exécutez les exits d'extrémité source. Il n'y a pas de statut d'échec pour ces exits.
9. Si le transfert aboutit, exécutez l'appel post-source (s'il est présent). Si l'appel post-source échoue, le transfert échoue.

## Exemples

Cet exemple illustre un déplacement de fichier de base entre agent1 et agent2. La commande permettant de démarrer le déplacement de fichier est envoyée à un gestionnaire de files d'attente appelé qm0, à l'aide d'une connexion en mode transport client. Le résultat de l'opération de transfert de fichier est affecté à la propriété appelée `move.result`.

```
<fte:filemove cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
  src="agent1@qm1" dst="agent2@qm2"
  rcproperty="move.result">

  <fte:filespec srcfilespec="/home/fteuser1/file.bin" dstfile="/home/fteuser2/file.bin"/>
</fte:filemove>
```

## Concepts associés

[Option de délai d'attente pour les transferts de fichiers en cours de récupération](#)

## Tâches associées

[Utilisation de Apache Ant avec MFT](#)

## tâche fte: ignoreissue Ant

Ignorez le résultat d'une commande **fte:filecopy**, **fte:filemove** ou **fte:call**. Lorsque vous spécifiez une tâche **fte:filecopy**, **fte:filemove** ou **fte:call** pour que le résultat soit `defer`, la tâche Ant alloue des ressources pour le suivi de ce résultat. Si le résultat ne vous intéresse plus, vous pouvez utiliser la tâche **fte:ignoreoutcome** pour libérer ces ressources.

## Attributs

### id

Obligatoire. Identifie le résultat qui n'a plus d'intérêt. En règle générale, vous spécifiez cet identificateur à l'aide d'une propriété que vous définissez à l'aide de l'attribut `idproperty` de la tâche [«tâche fte: filecopy Ant»](#), à la page 2200, [«fte: tâche Ant filemove»](#), à la page 2204 ou [«fte: appel de la tâche Ant»](#), à la page 2197.

## Exemple

Cet exemple montre comment utiliser la tâche `fte: ignoreissue` pour libérer les ressources allouées au suivi du résultat de la tâche [«tâche fte: filecopy Ant»](#), à la page 2200 précédente.

```
<!-- issue a file copy request -->
<fte:filecopy cmdqm="qm1@localhost@1414@SYSTEM.DEF.SVRCONN"
  src="agent1@qm1" dst="agent1@qm1"
  idproperty="copy.id"
  outcome="defer"/>

<!-- do some other things -->

<!-- decide that the result of the copy is not interesting -->
<fte:ignoreoutcome id="{copy.id}"/>
```

## Tâches associées

[Utilisation de Apache Ant avec MFT](#)

## tâche fte: ping Ant

Cette tâche IBM MQ Managed File Transfer Ant envoie une commande ping à un agent pour obtenir une réponse et détermine ainsi si l'agent peut traiter les transferts.

**Remarque :** IBM WebSphere MQ File Transfer Edition (FTE) n'est plus un produit pris en charge. Pour migrer de FTE vers le composant Managed File Transfer dans IBM MQ, voir [Migrating Managed File Transfer](#).



## Attributs

### agent

Obligatoire. Indique l'agent sur lequel la demande **fte:ping** doit être soumise. La valeur se présente sous la forme suivante: *agentname@qmgrname*, où *agentname* est le nom de l'agent et *qmgrname* est le nom du gestionnaire de files d'attente auquel cet agent est directement connecté.

### cmdqm

Facultatif. Gestionnaire de files d'attente de commandes auquel soumettre la demande. Spécifiez ces informations sous la forme *qmgrname@host@port@channel*, où:

- *qmgrname* est le nom du gestionnaire de files d'attente
- *host* est le nom d'hôte facultatif du système sur lequel le gestionnaire de files d'attente est en cours d'exécution.
- *port* est le numéro de port facultatif sur lequel le gestionnaire de files d'attente est en mode écoute
- *channel* est le canal SVRCONN facultatif à utiliser

Si vous omettez les informations *host*, *port* ou *channel* pour le gestionnaire de files d'attente de commandes, les informations de connexion spécifiées dans le fichier `command.properties` sont utilisées.



**Avertissement** : Si aucune valeur n'est spécifiée pour:

- Variable *host*, le mode de liaison est utilisé
- Variable *port*, la valeur 1414 est utilisée
- Variable *channel*, SYSTEM.DEF.SVRCONN est utilisé.

Voir [Le fichier MFT command.properties](#) pour plus d'informations.

Toutefois, vous ne pouvez pas ignorer les attributs du milieu, par exemple, *qmgrname@host@@channel*. Vous pouvez, par exemple, avoir *qmgrname@host*, ou *qmgrname@host@port*, ou *qmgrname@hostport@@channel*.

MFT fractionne l'attribut donné à l'aide du délimiteur @. En fonction du nombre de jetons trouvés, il prend le premier jeton comme *qmgrname*, le second comme *host*, le troisième comme *port* et enfin le *canal*.

Pour plus d'informations, voir [Le fichier MFT command.properties](#).

Vous pouvez utiliser la propriété **com.ibm.wmqfte.propertySet** pour spécifier le fichier `command.properties` à utiliser. Pour plus d'informations, voir [com.ibm.wmqfte.propertySet](#).

Si vous n'utilisez pas l'attribut `cmdqm`, la tâche utilise par défaut la propriété `com.ibm.wmqfte.ant.commandQueueManager`, si cette propriété est définie. Si la propriété `com.ibm.wmqfte.ant.commandQueueManager` n'est pas définie, une tentative de connexion au gestionnaire de files d'attente par défaut, défini dans le fichier `command.properties`, est effectuée. Le format de la propriété `com.ibm.wmqfte.ant.commandQueueManager` est identique à l'attribut `cmdqm`, c'est-à-dire *qmgrname@host@port@channel*.

### rcproperty

Obligatoire. Nomme une propriété dans laquelle stocker le code retour de l'opération **ping**.

### dépassement du délai d'attente

Facultatif. Durée maximale, en secondes, pendant laquelle la tâche attend la réponse de l'agent. Le délai d'attente minimal est de zéro seconde, mais un délai d'attente de moins un peut également être spécifié de sorte que la commande attend indéfiniment que l'agent réponde. Si aucune valeur n'est spécifiée pour `timeout`, la valeur par défaut est d'attendre jusqu'à 5 secondes que l'agent réponde.

## Exemple

Cet exemple envoie une demande **fte:ping** à agent1 hébergé par qm1. La demande **fte:ping** attend 15 secondes que l'agent réponde. Le résultat de la demande **fte:ping** est stocké dans une propriété appelée ping.rc.

```
<fte:ping agent="agent1@qm1" rcproperty="ping.rc" timeout="15"/>
```

## Codes retour

**0**

La commande a abouti.

**2**

La commande a expiré.

## Tâches associées

[Utilisation de Apache Ant avec MFT](#)

## tâche fte: uuid Ant

Génère un identificateur unique pseudo-aléatoire et l'affecte à une propriété donnée. Par exemple, vous pouvez utiliser cet identificateur pour générer des noms de travail pour d'autres opérations de transfert de fichiers.

## Attributs

### longueur

Obligatoire. Longueur numérique de l'identificateur unique universel à générer. Cette valeur de longueur n'inclut pas la longueur d'un préfixe, spécifiée par le paramètre **prefix**.

### property

Obligatoire. Nom de la propriété à laquelle affecter l'identificateur unique universel généré.

### préfixe

Facultatif. Préfixe à ajouter à l'identificateur unique universel généré. Ce préfixe n'est pas compté comme faisant partie de la longueur de l'UUID, comme indiqué par le paramètre **length**.

## Exemple

Cet exemple définit un UUID qui commence par les lettres ABC suivies de 16 caractères hexadécimaux pseudo-aléatoires. L'identificateur unique universel est affecté à une propriété nommée uuid.property.



```
<fte:uuid length="16" property="uuid.property" prefix="ABC"/>
```

## Tâches associées

[Utilisation de Apache Ant avec MFT](#)

## fte: filespec Ant élément imbriqué

Le paramètre **fte:filespec** est utilisé en tant qu'élément imbriqué dans d'autres tâches. Utilisez

**fte:filespec** pour décrire un mappage entre un ou plusieurs fichiers source, répertoires  ou ensembles de données et une destination. Généralement, cet élément est utilisé lors de l'expression d'un ensemble de fichiers ou de répertoires  ou d'ensembles de données à déplacer ou à copier.

## Imbriqué par:

- La tâche [fte: filecopy](#)

- Tâche `fte: filemove`

## Attributs de spécification source

Vous devez spécifier `srcfilespec` ou `srcqueue`.

### **spéc\_fichier\_source**

Indique la source de l'opération de fichier. La valeur de cet attribut peut inclure un caractère générique.

### **file\_src**

Indique que la source du transfert est une file d'attente. Le transfert déplace les données des messages stockés dans la file d'attente spécifiée par cet attribut. Vous ne pouvez pas spécifier cet attribut si la tâche **fte: filespec** est imbriquée dans la tâche **fte: filecopy**.

L'attribut `srcqueue` n'est pas pris en charge lorsque l'agent source est un agent de pont de protocole.

## Attributs de spécification de destination

Vous devez spécifier `dstdir`, `dstds`, `dstfilespace`, `dstfile`, `dstqueue` ou `dstpds`.

### **dstdir**

Indique un répertoire comme destination d'une opération de fichier.

### **dstds**

Indique un fichier comme destination d'une opération de fichier.

Cet attribut est pris en charge uniquement lorsque l'agent de destination est en cours d'exécution sur la plateforme z/OS.

### **dstfile**

Indique un fichier comme destination d'une opération de fichier.

### **espace fichier dstl**

Indique un espace fichier comme destination d'une opération de fichier.

Cet attribut s'applique uniquement si l'agent de destination est un agent Web IBM MQ 8.0 ayant accès à l'espace fichier de la passerelle Web.

### **dstpds**

Indique un fichier partitionné comme destination d'une opération de fichier.

Cet attribut est pris en charge uniquement lorsque l'agent de destination est en cours d'exécution sur la plateforme z/OS.

## file d'attente de données

Indique une file d'attente comme destination d'une opération de fichier à message. Vous pouvez éventuellement inclure un nom de gestionnaire de files d'attente dans cette spécification, en utilisant le format `QUEUE@QUEUEMANAGER`. Si vous n'indiquez pas de nom de gestionnaire de files d'attente, le gestionnaire de files d'attente de l'agent de destination est utilisé si vous n'avez pas défini la propriété d'agent de sortie `enableClusterQueueInputs` sur `true`. Si la propriété `enableClusterQueueInputOutput` est définie sur `true`, l'agent de destination utilise des procédures IBM MQ standard pour déterminer où se trouve la file d'attente. Vous devez indiquer un nom de file d'attente valide existant sur le gestionnaire de files d'attente.

Si vous spécifiez l'attribut `dstqueue`, vous ne pouvez pas spécifier les attributs `srcqueue` car ces attributs s'excluent mutuellement.

L'attribut `dstqueue` n'est pas pris en charge lorsque l'agent de destination est un agent de pont de protocole.

## Attributs de l'option source

### srcencoding

Facultatif. Codage de jeu de caractères utilisé par le fichier à transférer.

Vous pouvez spécifier cet attribut uniquement lorsque l'attribut `conversion` est défini sur la valeur `text`.

Si vous ne spécifiez pas l'attribut `srcencoding`, le jeu de caractères du système source est utilisé pour les transferts de texte.

### srceol

Facultatif. Délimiteur de fin de ligne utilisé par le fichier en cours de transfert. Les valeurs admises sont les suivantes :

- `CRLF` -Utilisez un caractère de retour chariot suivi d'un caractère de saut de ligne comme délimiteur de fin de ligne. Cette convention est typique des systèmes Windows .
- `LF` -Utilisez un caractère de saut de ligne comme délimiteur de fin de ligne. Cette convention est typique des systèmes UNIX .

Vous pouvez spécifier cet attribut uniquement lorsque l'attribut `conversion` est défini sur la valeur `text`. Si vous ne spécifiez pas l'attribut `srceol`, les transferts de texte déterminent automatiquement la valeur correcte en fonction du système d'exploitation de l'agent source.

### **srckeeptrailingspaces**

Facultatif. Détermine si les espaces de fin sont conservés dans les enregistrements source lus à partir d'un jeu de données de longueur fixe dans le cadre d'un transfert en mode texte. Les valeurs admises sont les suivantes :

- `true` -les espaces de fin sont conservés.
- `false` -les espaces de fin sont supprimés.

Si vous ne spécifiez pas l'attribut `srckeeptrailingspaces`, la valeur par défaut `false` est spécifiée.

Vous pouvez spécifier cet attribut uniquement si vous spécifiez également l'attribut `srcfilespec` et que vous affectez à l'attribut `conversion` la valeur `text`.

### srcmsgdelimbytes

Facultatif. Indique une ou plusieurs valeurs d'octet à insérer comme délimiteur lors de l'ajout de plusieurs messages à un fichier binaire. Chaque valeur doit être spécifiée sous la forme de deux chiffres hexadécimaux dans la plage 00-FF, préfixés par `x`. Plusieurs octets doivent être séparés par des virgules. Par exemple, `srcmsgdelimbytes="x08,xA4"`. Vous ne pouvez spécifier l'attribut `srcmsgdelimbytes` que si vous avez également spécifié l'attribut `srcqueue`. Vous ne pouvez pas spécifier l'attribut `srcmsgdelimbytes` si vous avez également spécifié la valeur `text` pour l'attribut `conversion`.

### srcmsgdelimtext

Facultatif. Indique une séquence de texte à insérer comme délimiteur lors de l'ajout de plusieurs messages à un fichier texte. Vous pouvez inclure des séquences d'échappement Java pour les littéraux chaîne dans le délimiteur. Par exemple, `srcmsgdelimtext="\u007d\n"`. Le délimiteur de texte est inséré après chaque message par l'agent source. Le délimiteur de texte est codé au format binaire à l'aide du codage source du transfert. Chaque message est lu au format binaire, le délimiteur codé est ajouté au message au format binaire et le résultat est transféré au format binaire à l'agent de destination. Si la page de codes de l'agent source inclut des états d'arrivée et de sortie, l'agent suppose que chaque message est à l'état de sortie à la fin du message. Au niveau de l'agent de destination, les données binaires sont converties de la même manière qu'un transfert de texte de fichier à fichier. Vous ne pouvez spécifier l'attribut `srcmsgdelimtext` que si vous avez également spécifié l'attribut `srcqueue` et la valeur `text` pour l'attribut `conversion`.

### srcmsgdelimposition

Facultatif. Indique la position dans laquelle le texte ou le délimiteur binaire est inséré. Les valeurs admises sont les suivantes :

- **prefix** -les délimiteurs sont insérés dans le fichier de destination avant les données de chaque message.
- **postfix** -les délimiteurs sont insérés dans le fichier de destination après les données de chaque message.

Vous pouvez spécifier l'attribut `srcmsgdelimposition` uniquement si vous avez également spécifié l'un des attributs `srcmsgdelimbytes` ou `srcmsgdelimtext` .

### **srcmsggroups**

Facultatif. Indique que les messages sont regroupés par ID groupe IBM MQ . Le premier groupe complet est écrit dans le fichier de destination. Si cet attribut n'est pas spécifié, tous les messages de la file d'attente source sont écrits dans le fichier de destination. Vous pouvez spécifier l'attribut `srcmsggroups` uniquement si vous avez également spécifié l'attribut `srcqueue` .

### **srcqueuetimeout**

Facultatif. Indique le délai d'attente, en secondes, pour que l'une des conditions suivantes soit remplie:

- Pour qu'un nouveau message soit écrit dans la file d'attente.
- Si l'attribut `srcmsggroups` a été spécifié, pour qu'un groupe complet soit écrit dans la file d'attente.

Si aucune de ces conditions n'est remplie dans le délai spécifié par la valeur de `srcqueuetimeout`, l'agent source arrête la lecture à partir de la file d'attente et termine le transfert. Si l'attribut `srcqueuetimeout` n'est pas spécifié, l'agent source arrête de lire la file d'attente source immédiatement si la file d'attente source est vide ou, dans le cas où l'attribut `srcmsggroups` est spécifié, s'il n'y a pas de groupe complet dans la file d'attente. Vous pouvez spécifier l'attribut `srcqueuetimeout` uniquement si vous avez également spécifié l'attribut `srcqueue` .

Pour plus d'informations sur la définition de la valeur `srcqueuetimeout` , voir [Conseils pour la spécification d'un temps d'attente sur un transfert de message vers fichier](#).

### **z/OS srcrcdelimbytes**

Facultatif. Indique une ou plusieurs valeurs d'octet à insérer comme délimiteur lors de l'ajout de plusieurs enregistrements d'un fichier source orienté enregistrement à un fichier binaire. Vous devez spécifier chaque valeur sous la forme de deux chiffres hexadécimaux dans la plage 00-FF, précédés de x. Plusieurs octets doivent être séparés par des virgules. Exemple :

```
srcrcdelimbytes="x08,xA4"
```

Vous pouvez spécifier l'attribut `srcrcdelimbytes` uniquement si le fichier source de transfert est orienté enregistrement, par exemple un ensemble de données z/OS , et si le fichier cible est un fichier normal, non orienté enregistrement. Vous ne pouvez pas spécifier l'attribut `srcrcdelimbytes` si vous avez également spécifié la valeur `text` pour l'attribut `conversion` .

### **srcrcdelimpos**

Facultatif. Indique la position dans laquelle le délimiteur binaire est inséré. Les valeurs admises sont les suivantes :

- **prefix**-les délimiteurs sont insérés dans le fichier de destination avant les données de chaque enregistrement de fichier orienté enregistrement source.
- **postfix**-les délimiteurs sont insérés dans le fichier de destination après les données de chaque enregistrement de fichier orienté enregistrement source.

Vous pouvez spécifier l'attribut `srcrcdelimpos` uniquement si vous avez également spécifié l'attribut `srcrcdelimbytes` .

## **Attributs de l'option de destination**

### **dstcodage**

Facultatif. Codage de jeu de caractères à utiliser pour le fichier transféré.

Vous pouvez spécifier cet attribut uniquement lorsque l'attribut `conversion` est défini sur la valeur `text`.

Si l'attribut `dstencoding` n'est pas spécifié, le jeu de caractères du système de destination est utilisé pour les transferts de texte.

#### **dsteol**

Facultatif. Délimiteur de fin de ligne à utiliser pour le fichier transféré. Les valeurs admises sont les suivantes :

- `CRLF` -Utilisez un caractère de retour chariot suivi d'un caractère de saut de ligne comme délimiteur de fin de ligne. Cette convention est typique des systèmes Windows .
- `LF` -Utilisez un caractère de saut de ligne comme délimiteur de fin de ligne. Cette convention est typique des systèmes UNIX .

Vous pouvez spécifier cet attribut uniquement lorsque l'attribut `conversion` est défini sur la valeur `text`.

Si vous ne spécifiez pas l'attribut `dsteol`, les transferts de texte déterminent automatiquement la valeur correcte en fonction du système d'exploitation de l'agent de destination.

#### **dstmsgdelimbytes**

Facultatif. Indique le délimiteur hexadécimal à utiliser lors du fractionnement d'un fichier binaire en plusieurs messages. Tous les messages ont le même ID groupe IBM MQ ; le dernier message du groupe a l'indicateur IBM MQ `LAST_MSG_IN_GROUP` défini. Le format de spécification d'un octet hexadécimal comme délimiteur est `xNN`, où N est un caractère compris entre 0 et 9 ou a-f. Vous pouvez spécifier une séquence d'octets hexadécimaux comme délimiteur en spécifiant une liste d'octets hexadécimaux séparés par des virgules, par exemple: `x3e,x20,x20,xbf`.

Vous pouvez spécifier l'attribut `dstmsgdelimbytes` uniquement si vous avez également spécifié l'attribut `dstqueue` et que le transfert est en mode binaire. Vous ne pouvez spécifier qu'un seul des attributs `dstmsgsize`, `dstmsgdelimbyteset` et `dstmsgdelimpattern`.

#### **dstmsgdelimpattern**

Facultatif. Indique l'expression régulière Java à utiliser lors du fractionnement d'un fichier texte en plusieurs messages. Tous les messages ont le même ID groupe IBM MQ ; le dernier message du groupe a l'indicateur IBM MQ `LAST_MSG_IN_GROUP` défini. Le format de spécification d'une expression régulière comme délimiteur est une expression régulière entre parenthèses, (*regular\_expression*) ou entre guillemets, "*regular\_expression*". Pour plus d'informations, voir [Expressions régulières utilisées par MFT](#).

Par défaut, la longueur de la chaîne à laquelle l'expression régulière peut correspondre est limitée par l'agent de destination à cinq caractères. Vous pouvez modifier ce comportement à l'aide de la propriété d'agent `maxDelimiterMatchLength`. Pour plus d'informations, voir [Propriétés de l'agent avancé MFT](#).

Vous pouvez spécifier l'attribut `dstmsgdelimpattern` uniquement si vous avez également spécifié l'attribut `dstqueue` et que le transfert est en mode texte. Vous ne pouvez spécifier qu'un seul des attributs `dstmsgsize`, `dstmsgdelimbyteset` et `dstmsgdelimpattern`.

#### **dstmsgdelimposition**

Facultatif. Indique la position dans laquelle le texte ou le délimiteur binaire doit se trouver. Les valeurs admises sont les suivantes :

- `prefix` -Les délimiteurs sont attendus au début de chaque ligne.
- `postfix` -Les délimiteurs sont attendus à la fin de chaque ligne.

Vous pouvez spécifier l'attribut `dstmsgdelimposition` uniquement si vous avez également spécifié l'attribut `dstmsgdelimpattern`.

#### **dstmsgincludedelim**

Facultatif. Indique si le délimiteur utilisé pour fractionner le fichier en plusieurs messages doit être inclus dans les messages. Si l'attribut `dstmsgincludedelim` est spécifié, le délimiteur est inclus à la fin du message qui contient les données de fichier précédant le délimiteur. Par défaut, le délimiteur

n'est pas inclus dans les messages. Vous pouvez spécifier l'attribut `dstmsgincludedelim` uniquement si vous avez également spécifié l'un des attributs `dstmsgdelimpattern` et `dstmsgdelimbytes`.

### **dstmsgpersistent**

Facultatif. Indique si les messages écrits dans la file d'attente de destination sont persistants. Les valeurs admises sont les suivantes :

- `true` -Ecrit les messages persistants dans la file d'attente de destination. Il s'agit de la valeur par défaut.
- `false` -Ecrit des messages non persistants dans la file d'attente de destination.
- `qdef` -La valeur de persistance est extraite de l'attribut `DefPersistence` de la file d'attente de destination.

Vous pouvez spécifier cet attribut uniquement lorsque l'attribut `dstqueue` est également spécifié.

### **dstmsgprops**

Facultatif. Indique si les propriétés de message IBM MQ sont définies pour le premier message écrit dans la file d'attente de destination par le transfert. Les valeurs possibles sont les suivantes :

- `true` -Définit les propriétés de message sur le premier message créé par le transfert.
- `false` -Ne pas définir de propriétés de message sur le premier message créé par le transfert. Il s'agit de la valeur par défaut.

Pour plus d'informations, voir [Propriétés des messagesMQ définies par MFT sur les messages écrits dans les files d'attente de destination](#).

Vous pouvez spécifier cet attribut uniquement lorsque l'attribut `dstqueue` est également spécifié.

### **taille\_msgdst**

Facultatif. Indique si le fichier doit être fractionné en plusieurs messages de longueur fixe. Tous les messages ont le même ID groupe IBM MQ ; l'indicateur IBM MQ `LAST_MSG_IN_GROUP` est défini pour le dernier message du groupe. La taille des messages est spécifiée par la valeur de `dstmsgsize`. Le format de `dstmsgsize` est *longueurunités*, où *longueur* est un entier positif et *unités* est l'une des valeurs suivantes:

- `B` -Octets. La valeur minimale autorisée est deux fois la valeur maximale en octets par caractère de la page de codes des messages de destination.
- `K` -Kibiocets. Cela équivaut à 1024 octets.
- `M` -Mébioctets. Cela équivaut à 1024 kibiocets.

Si le fichier est transféré en mode texte et se trouve dans un jeu de caractères codés sur deux octets ou dans un jeu de caractères codés sur plusieurs octets, il est scindé en messages sur la limite de caractères la plus proche de la taille de message spécifiée.

Vous pouvez spécifier l'attribut `dstmsgsize` uniquement si vous avez également spécifié l'attribut `dstqueue`. Vous ne pouvez spécifier qu'un seul des attributs `dstmsgsize`, `dstmsgdelimbyteset` et `dstmsgdelimpattern`.

### **dstunsupportedcodepage**

Facultatif. Indique l'action à effectuer si le gestionnaire de files d'attente de destination, tel que spécifié par l'attribut `dstqueue`, ne prend pas en charge la page de codes utilisée lors du transfert de données de fichier vers une file d'attente en tant que transfert de texte. Les valeurs admises pour cet attribut sont les suivantes:

- `binary` -poursuit le transfert mais n'applique pas la conversion de page de codes aux données en cours de transfert. La spécification de cette valeur revient à ne pas définir l'attribut de conversion sur `text`.
- `fail` -ne pas poursuivre l'opération de transfert. Le fichier est enregistré comme n'ayant pas pu être transféré. Il s'agit de l'option par défaut.

Vous ne pouvez spécifier l'attribut `dstunsupportedcodepage` que si vous avez également spécifié l'attribut `dstqueue` et la valeur `text` pour l'attribut `conversion`.

## ► z/OS **dsttruncaterecords**

Facultatif. Indique que les enregistrements de destination dont la longueur est supérieure à l'attribut de fichier LRECL sont tronqués. Si la valeur est true, les enregistrements sont tronqués. Si la valeur est false, les enregistrements sont encapsulés. Par défaut, le paramètre a la valeur false. Ce paramètre est valide uniquement pour les transferts en mode texte où la destination est un fichier.

## Autres attributs

### total de contrôle

Facultatif. Détermine l'algorithme utilisé pour la somme de contrôle des fichiers transférés.

- MD5 -utilisez l'algorithme de hachage MD5 .
- NONE -n'utilisez pas d'algorithme de total de contrôle.

Si vous ne spécifiez pas l'attribut checksum , la valeur par défaut MD5 est utilisée.

### conversion

Facultatif. Indique le type de conversion à appliquer au fichier lors de son transfert. Les valeurs possibles sont les suivantes :

- binary -n'applique aucune conversion.
- text -applique la conversion de page de codes entre les systèmes source et cible. Appliquez également la conversion des délimiteurs de ligne. Les attributs srcencoding, dstencoding, srceol et dsteol influencent la conversion appliquée.

Si vous ne spécifiez pas l'attribut conversion , la valeur par défaut binary est spécifiée.

### Ecraser

Facultatif. Détermine si un fichier de destination existant ► z/OS ou un fichier peut être écrasé par l'opération. Lorsque vous spécifiez la valeur true, les fichiers de destination ► z/OS ou les fichiers existants sont écrasés. Lorsque vous spécifiez la valeur false, l'existence d'un fichier en double ► z/OS ou d'un fichier à la destination entraîne l'échec de l'opération. Si l'attribut overwrite n'est pas spécifié, la valeur par défaut false est spécifiée.

### exécuter de façon récursive

Facultatif. Détermine si le transfert de fichiers est récursif dans les sous-répertoires. Lorsque vous spécifiez la valeur true, le transfert est récursif dans les sous-répertoires. Lorsque vous spécifiez la valeur false, le transfert n'est pas récursif dans les sous-répertoires. Si l'attribut recurse n'est pas spécifié, la valeur par défaut false est spécifiée.

## Exemple

Cet exemple spécifie fte: filespec avec un fichier source file1.bin et un fichier de destination file2.bin.

```
<fte:filespec srcfilespec="/home/fteuser/file1.bin" dstfile="/home/fteuser/file2.bin"/>
```

## Tâches associées

[Utilisation de Apache Ant avec MFT](#)

## Éléments imbriqués fte: metadata Ant

Les métadonnées sont utilisées pour transmettre des informations supplémentaires définies par l'utilisateur avec une opération de transfert de fichiers.

Pour plus d'informations sur la façon dont Managed File Transfer utilise les métadonnées, voir «[Métadonnées pour les exits utilisateur MFT](#)», à la page 2221 .



## Imbriqué par:

- La tâche [fte: filecopy](#)
- Tâche [fte: filemove](#)
- La tâche [fte: call](#)

## Paramètres spécifiés en tant qu'éléments imbriqués

### fte: entrée

Vous devez spécifier au moins une entrée dans l'élément imbriqué `fte:metadata` . Vous pouvez choisir d'indiquer plusieurs entrées. Les entrées associent un nom de clé à une valeur. Les clés doivent être uniques dans un bloc de `fte:metadata`

## Attributs d'entrée

### name

Obligatoire. Nom de la clé appartenant à cette entrée. Ce nom doit être unique dans tous les paramètres **entry** imbriqués dans un élément `fte: metadata` .

### valeur

Obligatoire. Valeur à affecter à cette entrée.

## Exemple

Cet exemple illustre une définition `fte:metadata` qui contient deux entrées.

```
<fte:metadata>
  <fte:entry name="org.foo.partColor" value="red"/>
  <fte:entry name="org.foo.partSize" value="medium"/>
</fte:metadata>
```

## Tâches associées

[Utilisation de Apache Ant avec MFT](#)

## Éléments imbriqués d'appel de programme

Les programmes peuvent être démarrés à l'aide de l'un des cinq éléments imbriqués: `fte:presrc`, `fte:predst`, `fte:postdst`, `fte:postsrct` et `fte:command`. Ces éléments imbriqués demandent à un agent d'appeler un programme externe dans le cadre de son traitement. Avant de démarrer un programme, vous devez vous assurer que la commande se trouve dans un emplacement spécifié par la propriété `commandPath` dans le fichier `agent.properties` de l'agent qui exécute la commande.

Même si chaque élément d'appel de programme a un nom différent, ils partagent le même ensemble d'attributs et le même ensemble d'éléments imbriqués. Les programmes peuvent être démarrés par les tâches Ant **fte:filecopy**, **fte:filemove** et **fte:command** .

Vous ne pouvez pas appeler de programmes à partir d'un agent de pont `Connect:Direct` .

## Tâches Ant pouvant appeler des programmes:

- La tâche [fte: filecopy](#) imbrique les paramètres d'appel de programme à l'aide des éléments imbriqués `fte:predst`, `fte:postdst`, `fte:presrct` et `fte:postsrct` .
- La tâche [fte: filemove](#) imbrique les paramètres d'appel de programme à l'aide des éléments imbriqués `fte:predst`, `fte:postdst`, `fte:presrct` et `fte:postsrct` .
- La tâche [fte: call](#) imbrique les paramètres d'appel de programme à l'aide de l'élément imbriqué `fte:command` .

## Attributs

### commande

Obligatoire. Nomme le programme à appeler. Pour que l'agent puisse exécuter une commande, celle-ci doit se trouver dans un emplacement spécifié par la propriété `commandPath` dans le fichier `agent.properties` de l'agent. Pour plus d'informations, voir [commandPath MFT property](#). Toute information de chemin spécifiée dans l'attribut `command` est considérée comme relative à un emplacement spécifié par la propriété `commandPath`. Lorsque `type` est `executable`, un programme exécutable est attendu, sinon un script approprié au type d'appel est attendu.

### Nombre de relances

Facultatif. Nombre de tentatives d'appel du programme si le programme ne renvoie pas de code retour de réussite. Le programme nommé par l'attribut `command` est appelé jusqu'à ce nombre de fois. La valeur affectée à cet attribut doit être non négative. Si vous ne spécifiez pas l'attribut `retrycount`, la valeur par défaut zéro est utilisée.

### attente de nouvelle tentative

Facultatif. Délai d'attente, en secondes, avant une nouvelle tentative d'appel du programme. Si le programme nommé par l'attribut `command` ne renvoie pas de code retour de réussite et que l'attribut `retrycount` indique une valeur différente de zéro, ce paramètre détermine le délai d'attente entre les nouvelles tentatives. La valeur affectée à cet attribut doit être non négative. Si vous ne spécifiez pas l'attribut `retrywait`, la valeur par défaut zéro est utilisée.

### Code de réussite

Facultatif. La valeur de cet attribut est utilisée pour déterminer à quel moment l'appel de programme s'exécute correctement. Le code retour du processus pour la commande est évalué à l'aide de cette expression. La valeur peut être composée d'une ou de plusieurs expressions combinées avec un caractère de barre verticale (|) pour signifier la valeur booléenne OR ou une perluète (&) pour indiquer la valeur booléenne AND. Chaque expression peut être l'un des types d'expression suivants:

- Nombre indiquant un test d'égalité entre le code retour du processus et le nombre.
- Nombre précédé du caractère ">" pour indiquer un test supérieur à entre le nombre et le code retour du processus.
- Nombre précédé du caractère "<" pour indiquer un test inférieur à entre le nombre et le code retour du processus.
- Un nombre précédé d'un "!" pour indiquer un test non égal à entre le numéro et le code retour du processus.

Par exemple, `>2&<7&!5|0|14` est interprété comme les codes retour suivants ayant abouti: 0, 3, 4, 6, 14. Tous les autres codes retour sont interprétés comme n'ayant pas abouti. Si vous ne spécifiez pas l'attribut `successrc`, la valeur par défaut zéro est utilisée. Cela signifie que la commande est considérée comme ayant abouti si, et seulement si, elle renvoie un code de zéro.

### type

Facultatif. La valeur de cet attribut indique le type de programme appelé. Indiquez l'une des options suivantes :

#### exécutable

La tâche appelle un programme exécutable. Peut avoir des arguments supplémentaires spécifiés à l'aide de l'élément imbriqué `arg`. Le programme doit être accessible sur le `commandPath` et, le cas échéant, disposer du droit d'exécution défini. Les scripts UNIX peuvent être appelés tant qu'ils spécifient un programme shell (par exemple, la première ligne du fichier script shell est: `#!/bin/sh`). La sortie de la commande écrite dans `stderr` ou `stdout` est envoyée au journal Managed File Transfer pour l'appel. Toutefois, la quantité de données générées est limitée par la configuration de l'agent. La valeur par défaut est 10K octets de données, mais vous pouvez remplacer cette valeur par défaut à l'aide de la propriété d'agent: `maxCommandOutput`.

#### antscript

La tâche exécute le script Ant spécifié à l'aide de la commande `fteAnt`. Les propriétés peuvent être spécifiées à l'aide de l'élément imbriqué `property`. Les cibles Ant peuvent être spécifiées à l'aide de l'élément imbriqué `target`. Le script Ant doit être accessible dans `commandPath`. La sortie Ant écrite dans `stderr` ou `stdout` est envoyée au journal Managed File Transfer pour l'appel.

Toutefois, la quantité de données générées est limitée par la configuration de l'agent. La valeur par défaut est 10K octets de données, mais vous pouvez la remplacer à l'aide de la propriété d'agent: `maxCommandOutput`.

## z/OS JCL

La valeur `jcl` est prise en charge sous z/OS uniquement et exécute le script JCL z/OS spécifié. Le JCL est soumis en tant que travail et requiert la présence de la carte de travail. Lorsque le travail est soumis avec succès, la sortie de la commande JCL, écrite dans le journal Managed File Transfer, contient le texte suivant: `JOB nom_travail(ID_travail)` où:

- `nom_travail` est le nom du travail identifié par la carte de travail dans le JCL.
- `job_id` est l'ID du travail généré par le système z/OS.

Si le travail ne peut pas être soumis correctement, la commande de script JCL échoue et écrit un message dans le journal indiquant la raison de l'échec (par exemple, aucune carte de travail n'est présente). Pour savoir si le travail a été exécuté ou exécuté avec succès, utilisez un service système tel que SDSF. Managed File Transfer ne fournit pas les informations car il soumet uniquement le travail; le système détermine ensuite quand exécuter le travail et comment la sortie du travail est présentée. Étant donné qu'un script JCL est soumis en tant que travail par lots, il n'est pas conseillé de spécifier `jcl` pour un élément imbriqué `presrc` ou `predst` car vous savez uniquement que le travail a été soumis avec succès et non pas s'il a été exécuté avec succès avant le début du transfert. Aucun élément imbriqué n'est valide avec le type `jcl`.

L'exemple suivant illustre un travail JCL:

```
//MYJOB JOB
//*
//MYJOB EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=H
//SYSUT1 DD DSN=FRED.DEMO.TXT,DISP=SHR
//SYSUT2 DD DSN=BOB.DEMO.TXT,DISP=(NEW,CATLG),
// RECFM=VB,LRECL=133,BLKSIZE=2048,
// SPACE=(TRK,(30,5),RLSE)
//SYSIN DD DUMMY
```

## Paramètres spécifiés en tant qu'éléments imbriqués

### **fte: arg**

Valide uniquement lorsque la valeur de l'attribut `type` est `executable`. Utilisez des éléments `fte: arg` imbriqués pour spécifier des arguments pour le programme appelé dans le cadre de l'appel de programme. Les arguments de programme sont générés à partir des valeurs spécifiées par les éléments `fte: arg` dans l'ordre dans lequel les éléments `fte: arg` sont détectés. Vous pouvez choisir de spécifier zéro ou plusieurs éléments `fte: arg` en tant qu'éléments imbriqués d'un appel de programme.

### **fte: propriété**

Valide uniquement lorsque la valeur de l'attribut `type` est `antscript`. Utilisez les attributs `name` et `value` des éléments `fte: property` imbriqués pour transmettre des paires nom-valeur au script Ant. Vous pouvez choisir de spécifier zéro ou plusieurs éléments `fte: property` en tant qu'éléments imbriqués d'un appel de programme.

### **fte: cible**

Valide uniquement lorsque la valeur de l'attribut `type` est `antscript`. Spécifiez une cible dans le script Ant à appeler. Vous pouvez choisir de spécifier zéro ou plusieurs éléments `fte: target` en tant qu'éléments imbriqués d'un appel de programme.

## Attributs d'argument

### **valeur**

Obligatoire. Valeur de l'argument à transmettre au programme appelé.

## Attributs de propriété

### nom

Obligatoire. Nom d'une propriété à transmettre au script Ant .

### valeur

Obligatoire. Valeur à associer au nom de propriété transmis au script Ant .

### Exemples

Cet exemple illustre un appel de programme `fte:postsrc` spécifié dans le cadre d'une tâche `fte:filecopy` . L'appel de programme est destiné à un programme appelé `post.sh` et est fourni avec un seul argument de `/home/fteuser2/file.bin` .

```
<fte:filecopy
cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
src="agent1@qm1" dst="agent2@qm2"
rcproperty="copy.result">
<fte:filespec srcfilespec="/home/fteuser1/file.bin" dstfile="/home/fteuser2/file.bin"/>
<fte:postsrc command="post.sh" successsrc="1" >
  <fte:arg value="/home/fteuser2/file.bin"/>
</fte:postsrc>
</fte:filecopy>
```

Cet exemple illustre un appel de programme `fte:command` spécifié dans le cadre d'une tâche `fte:call` . L'appel de programme est destiné à un exécutable appelé `command.sh` , qui ne transmet aucun argument de ligne de commande. Si `command.sh` ne renvoie pas le code retour de réussite 1, la commande est relancée au bout de 30 secondes.

```
<fte:call
cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
agent="agent1@qm1"
rcproperty="call.rc"
origuser="bob"
jobname="${job.id}">
<fte:command command="command.sh" successsrc="1" retrycount="5" retrywait="30"/>
</fte:call>
```

Cet exemple illustre un appel de programme `fte:command` spécifié dans le cadre d'une tâche `fte:call` . L'appel de programme concerne les cibles de copie et de compression dans un script Ant appelé `script.xml` , auquel deux propriétés sont transmises.

```
<fte:call
cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
agent="agent1@qm1"
rcproperty="call.rc"
origuser="bob"
jobname="${job.id}">
<fte:command command="script.xml" type="antscript">
  <property name="src" value="AGENT5@QM5"/>
  <property name="dst" value="AGENT3@QM3"/>
  <target name="copy"/>
  <target name="compress"/>
</fte:command>
</fte:call>
```

### Tâches associées

[Spécification des programmes à exécuter avec MFT](#)

[Utilisation de Apache Ant avec MFT](#)

## Exits utilisateur MFT pour la référence de personnalisation

Informations de référence pour vous aider à configurer des exits utilisateur pour Managed File Transfer.

### Concepts associés

[Exits utilisateur source et cible MFT](#)

## Métadonnées pour les exits utilisateur MFT

Il existe trois types de métadonnées qui peuvent être fournis aux routines d'exit utilisateur pour Managed File Transfer: métadonnées d'environnement, de transfert et de fichier. Ces métadonnées sont présentées sous forme de mappes de paires clé-valeur Java .

### Métadonnées d'environnement

Les métadonnées d'environnement sont transmises à toutes les routines d'exit utilisateur et décrivent l'environnement d'exécution de l'agent à partir duquel la routine d'exit utilisateur est appelée. Ces métadonnées sont en lecture seule et ne peuvent pas être mises à jour par une routine d'exit utilisateur.

Clé	Description
CLÉ_RÉPERTOIRE_CONFIGURATION_AGENT	Nom du répertoire contenant les informations de configuration de l'agent.
CLE_REPERTOIRE_DU PRODUIT	Nom du répertoire dans lequel le code d'agent a été installé.
CL_VERSION_AGENT_	Numéro de version de l'environnement d'exécution de l'agent qui appelle la routine d'exit.

Les noms de clé et les noms de valeur indiqués dans le tableau 1 sont des constantes qui sont définies dans l'interface EnvironmentMetaDataConstants .

### Métadonnées de transfert

Les métadonnées de transfert sont transmises à toutes les routines d'exit utilisateur. Les métadonnées sont constituées de valeurs fournies par le système et de valeurs fournies par l'utilisateur. Si vous modifiez des valeurs fournies par le système, ces modifications sont ignorées. Les valeurs initiales fournies par l'utilisateur pour l'exit utilisateur de début de transfert source sont basées sur les valeurs que vous fournissez lorsque vous définissez le transfert. L'agent source peut modifier les valeurs fournies par l'utilisateur dans le cadre du traitement de l'exit utilisateur de démarrage de transfert source. Cet exit utilisateur est appelé avant le début du transfert complet de fichiers. Ces modifications sont utilisées dans les appels ultérieurs à d'autres routines d'exit liées à ce transfert. Les métadonnées de transfert sont appliquées à un transfert complet.

Bien que tous les exits utilisateur puissent lire les valeurs des métadonnées de transfert, seul l'exit utilisateur de démarrage de transfert source peut modifier les métadonnées de transfert

Vous ne pouvez pas utiliser les métadonnées de transfert pour propager des informations entre différents transferts de fichiers.

Les métadonnées de transfert fournies par le système sont détaillées dans le tableau 2:

Clé	Description
CLÉ_AGENT_DESTINATION	Nom de l'agent qui est la destination du transfert.
CL_NOM_TRAVAIL	Nom de travail associé à la demande de transfert
CL_UTILISATEUR_MQDM	Zone utilisateur MQDM du message utilisé pour soumettre la demande de transfert
CLÉ_HÔTE_ORIGINE	Nom d'hôte spécifié comme nom d'hôte d'origine dans la demande de transfert
CL_UTILISATEUR_ORIGINATING_	Nom d'utilisateur spécifié comme ID utilisateur d'origine dans la demande de transfert

Tableau 883. Métadonnées de transfert (suite)

Clé	Description
CL_AGENT_SOURCE	Nom de l'agent qui est la source du transfert
CLÉ D'ID_TRANSACTION	Identificateur du transfert

Les noms de clé et les noms de valeur indiqués dans le tableau 2 sont des constantes définies dans l'interface TransferMetaDataConstants .

### Métadonnées de fichier

Les métadonnées de fichier sont transmises à l'exit de démarrage de transfert source dans le cadre de la spécification de fichier. Il existe des métadonnées de fichier distinctes pour les fichiers source et cible.

Vous ne pouvez pas utiliser les métadonnées de fichier pour propager des informations entre différents transferts de fichiers.

Tableau 884. Métadonnées de fichier

Clé	Valeurs admises	Description
Séparateurs CONVERT_LINE_SEPARATORS		Valeur clé utilisée pour les transferts de texte pour indiquer si les séquences de séparateurs de ligne CRLF (retour chariot-saut de ligne) ou LF (saut de ligne) dans les données source sont converties en séquence de séparateurs de ligne à la destination.
CL_DELIMITER_DE		Valeur de clé utilisée pour définir un délimiteur afin de séparer les données d'enregistrement lors du transfert de données orientées enregistrement vers des fichiers normaux.  Egalement utilisé pour les transferts de message à fichier et de fichier à message.
CLE_DELIMITER_POSITION_KEY	VALEUR_PREFIXE_POSITION_DELIMITER_POSITION_POS TFIX_VALUE	A utiliser avec DELIMITER_KEY pour définir la position du délimiteur (préfixe ou suffixe).
CLÉ_TYPE_DÉLIMITEUR	VALEUR_BINAIRE_TYPE_DÉLIMITEUR VALEUR_TEXTE_TYPE_DÉLIMITEUR VALEUR_TAILLE_TYPE_DÉLIMITEUR	A utiliser avec DELIMITER_KEY pour définir le type de délimiteur.
CLÉ_EXIST_DESTINATION	DESTINATION_EXIST_KEY_ERROR_VALUE DESTINATION_EXIST_KEY_OVERWRITE_VALUE	Détermine le comportement du transfert de fichier si le fichier de destination existe.
CL_ALIAS_FILE		Valeur de clé utilisée pour définir un alias pour le fichier en cours de transfert.
FILE_CHECKSUM_METHOD_KEY	FILE_CHECKSUM_METHOD_NONE_VALUE FILE_CHECKSUM_METHOD_MD5_VALUE	Détermine la méthode de total de contrôle à utiliser lors du transfert du fichier.
CL_CONVERSION_FILE_	VALEUR_TEXTE_CONVERSION_FICHIER VALEUR_CONVERSION_BINAIRE_FICHIER	Détermine le type de conversion appliqué au contenu du fichier.
CLÉ_ENCODING_FICHIER		Détermine le codage utilisé pour un fichier texte.
FILE_END_OF_LINE_KEY (clé de ligne)	FILE_END_OF_LINE_LF_VALUE FILE_END_OF_LINE_CRLF_VALUE	Détermine la séquence de caractères qui indique la fin d'une ligne: < LF > ou < CR> < LF>.

Tableau 884. Métadonnées de fichier (suite)		
Clé	Valeurs admises	Description
ALIAS_ESPACE_FICHER		Détermine l'alias d'un fichier dans l'espace fichier. <b>Remarque :</b> Ces métadonnées ne peuvent être utilisées que si FILE_TYPE_KEY est FILE_TYPE_FILE_SPACE_VALUE
NOM_ESPACE_FICHER		Détermine le nom de l'espace fichier. <b>Remarque :</b> Ces métadonnées ne peuvent être utilisées que si FILE_TYPE_KEY est FILE_TYPE_FILE_SPACE_VALUE
CLE_TYPE_FICHER	FILE_TYPE_FILE_VALUE FILE_TYPE_DIRECTORY_VALUE FILE_TYPE_DATASET_VALUE FILE_TYPE_PDS_VALUE FILE_TYPE_QUEUE_VALUE FILE_TYPE_FILE_SPACE_VALUE	Détermine la spécification du fichier, de la file d'attente ou de l'espace fichier cible.
CL_ID_GROUPE		Valeur de clé utilisée pour les transferts de message à fichier afin de déterminer le groupe de messages à lire à partir de la file d'attente source. Cet attribut est valide uniquement lorsque la valeur de USE_GROUPS_KEY est USE_GROUPS_TRUE_VALUE.
INCLUDE_DELIMITER_IN_MESSAGE_KEY	INCLUDE_DELIMITER_IN_MESSAGE_TRUE_VALUE INCLUDE_DELIMITER_IN_MESSAGE_FALSE_VALUE	Valeur de clé utilisée pour les transferts de fichier à message afin de déterminer s'il convient d'inclure les délimiteurs utilisés pour fractionner le fichier en plusieurs messages à la fin des messages. Cet attribut est valide uniquement lorsque la valeur de DELIMITER_TYPE_KEY est DELIMITER_TYPE_BINARY_VALUE DELIMITER_TYPE_TEXT_VALUE.
INSERT_RECORD_LINE_SEPARATOR_KEY		Valeur de clé utilisée pour les transferts de texte à partir de fichiers orientés enregistrement afin d'indiquer si des séparateurs de ligne sont insérés dans les données après chaque enregistrement.
CLÉ_ESPACE_DE_DÉBUT	KEEP_TRAILING_SPACES_TRUE_VALUE KEEP_TRAILING_SPACES_FALSE_VALUE	Valeur de clé utilisée pour déterminer si les espaces de fin sont supprimés des enregistrements lus à partir de fichiers de longueur fixe.
NOUVEAU_ENREGISTREMENT_SUR_LIGNE_CLÉ_SÉPARATEUR_CLÉ		Valeur clé utilisée pour les transferts de texte vers des fichiers orientés enregistrement afin d'indiquer si les séparateurs de ligne dans les données sont inclus avec les données d'enregistrement ou provoquent un nouvel enregistrement (et ne sont pas écrits).
CLE DE PERSISTANT	VALEUR_FICHER_PERSISTANT VALEUR DE PERSISTANT VALEUR PAR DEFAUT DE LA FILE D'ATTENTE	Valeur de clé utilisée pour les transferts de fichier à message afin de déterminer si les messages sont persistants.

Tableau 884. Métadonnées de fichier (suite)		
Clé	Valeurs admises	Description
CL_PROPS_MQ_SET_	VALEUR DE TRUE_SET_MQ_PROPS_ SET_MQ_PROPS_FALSE_VALUE	Valeur de clé utilisée pour les transferts de fichier à message afin de déterminer si les propriétés de message IBM MQ sont définies sur le premier message d'un fichier et sur tous les messages écrits dans la file d'attente en cas d'erreur.
CLÉ_PAGE_CODE_NON reconnaissable	VALEUR_ÉCHEC_PAGE_NON_RECONNAISSANCE VALEUR_ÉCHEC_CODE_PAGE_BINNAIRE_NON RECONNAISSABLE	Valeur de clé utilisée pour les transferts de fichier à message afin de déterminer si un transfert en mode texte échoue ou si la conversion est effectuée, si la page de codes des données n'est pas reconnue par le gestionnaire de files d'attente de destination.
CL_GROUPE_USE	USE_GROUPS_TRUE_VALUE VALEUR_UTILISATION_GROUPE_FALSE_VALUE	Valeur de clé utilisée pour les transferts de message à fichier afin de déterminer s'il convient de ne transférer qu'un groupe complet de messages de la file d'attente source.
CLE_WAIT_TIME_CLE		Valeur de clé utilisée pour les transferts de message à fichier afin de déterminer le délai d'attente, en secondes, de l'agent source pour l'un des cas suivants: <ul style="list-style-type: none"> <li>• Message à afficher dans la file d'attente source, si la file d'attente est vide ou est devenue vide, si la valeur de USE_GROUPS_KEY est FALSE.</li> <li>• Groupe complet à afficher dans la file d'attente source, si la valeur de USE_GROUPS_KEY est TRUE.</li> </ul>

Les noms de clé et les noms de valeur indiqués dans le tableau 3 sont des constantes définies dans l'interface FileMetaDataConstants .

### Concepts associés

«Interfaces Java pour les exits utilisateur MFT», à la page 2231

Utilisez les rubriques de cette section pour obtenir des informations de référence sur les interfaces Java pour les routines d'exit utilisateur.

### Tâches associées

Personnalisation de MFT avec des exits utilisateur

### Référence associée

«Exits utilisateur du moniteur de ressources MFT», à la page 2225

Les exits utilisateur du moniteur de ressources permettent de configurer le code personnalisé à exécuter lorsque la condition de déclenchement d'un moniteur est remplie, avant le démarrage de la tâche associée.

«Propriétés de l'agent MFT pour les exits utilisateur», à la page 2229

Outre les propriétés standard du fichier agent .properties , il existe plusieurs propriétés avancées spécifiques aux routines d'exit utilisateur. Ces propriétés ne sont pas incluses par défaut. Par conséquent, si vous souhaitez les utiliser, vous devez éditer manuellement le fichier agent .properties . Si vous modifiez le fichier agent .properties alors que cet agent est en cours d'exécution, arrêtez et redémarrez l'agent pour récupérer les modifications.



## Exits utilisateur du moniteur de ressources MFT

Les exits utilisateur du moniteur de ressources permettent de configurer le code personnalisé à exécuter lorsque la condition de déclenchement d'un moniteur est remplie, avant le démarrage de la tâche associée.

Il n'est pas recommandé d'appeler de nouveaux transferts directement à partir du code d'exit utilisateur. Dans certaines circonstances, les fichiers sont transférés plusieurs fois car les exits utilisateur ne sont pas résilients aux redémarrages d'agent.

Les exits utilisateur du moniteur de ressources utilisent l'infrastructure existante pour les exits utilisateur. Les exits utilisateur du moniteur sont appelés après le déclenchement d'un moniteur, mais avant l'exécution de la tâche correspondante par la tâche du moniteur. Cela permet à l'exit utilisateur de modifier la tâche à exécuter et de décider si une tâche doit être exécutée ou non. Vous pouvez modifier la tâche de surveillance en mettant à jour les métadonnées de surveillance, qui sont ensuite utilisées pour la substitution de variables dans le document de tâche créé par la création du moniteur d'origine. L'exit du moniteur peut également remplacer ou mettre à jour la chaîne XML de définition de tâche transmise en tant que paramètre. L'exit du moniteur peut renvoyer le code de résultat 'continue' ou 'cancel' pour la tâche. Si l'annulation est renvoyée, la tâche ne sera pas démarrée et le moniteur ne redémarrera pas tant que la ressource contrôlée ne correspondra pas aux conditions de déclenchement. Si la ressource n'a pas été modifiée, le déclencheur ne démarre pas. Comme pour les autres exits utilisateur, vous pouvez chaîner les exits de surveillance. Si l'un des exits renvoie un code de résultat d'annulation, le résultat global est l'annulation et la tâche n'est pas démarrée.

- Une mappe de métadonnées d'environnement (identique à d'autres exits utilisateur)
- Mappe de métadonnées de surveillance incluant des métadonnées système non modifiables et des métadonnées utilisateur modifiables. Les métadonnées système non modifiables sont les suivantes:
  - FILENAME-Nom du fichier qui a satisfait à la condition de déclenchement
  - FILEPATH-chemin d'accès au fichier qui satisfait à la condition de déclenchement
  - FILESIZE (en octets-ces métadonnées peuvent ne pas être présentes)-taille du fichier qui satisfait à la condition de déclenchement
  - LASTMODIFIEDDATE (Local)-date à laquelle le fichier qui satisfait à la condition de déclenchement a été modifié pour la dernière fois. Cette date est exprimée par rapport à la date locale du fuseau horaire de l'agent et est formatée sous forme de date ISO 8601.
  - LASTMODIFIEDTIME (Local)-heure au format local à laquelle le fichier qui satisfait la condition de déclenchement a été modifié pour la dernière fois. Cette heure est exprimée par rapport à l'heure locale du fuseau horaire de l'agent et est formatée sous forme d'heure ISO 8601.
  - LASTMODIFIEDDATEUTC-date au format universel que le fichier qui satisfait à la condition de déclenchement a été modifié pour la dernière fois. Cette date est exprimée sous forme de date locale convertie au fuseau horaire TUC et est formatée sous forme de date ISO 8601.
  - LASTMODIFIEDTIMEUTC-heure au format universel à laquelle le fichier qui satisfait la condition de déclenchement a été modifié pour la dernière fois. Cette heure est exprimée sous forme d'heure locale convertie au fuseau horaire TUC et est formatée sous forme d'heure ISO 8601.
  - AGENTNAME-nom de l'agent de surveillance
- Chaîne XML représentant la tâche à exécuter suite au déclenchement du moniteur.

Les exits de surveillance renvoient les données suivantes:

- Un indicateur qui indique si la progression doit se poursuivre (continuer ou annuler)
- Chaîne à insérer dans le message de journal de déclenchement satisfait

Suite à l'exécution du code d'exit du moniteur, les métadonnées du moniteur et la chaîne XML de définition de tâche qui ont été initialement transmises en tant que paramètres peuvent également avoir été mises à jour.

La valeur de la propriété d'agent `monitorExitClasses` (dans le fichier `agent.properties`) indique les classes d'exit de surveillance à charger, chaque classe d'exit étant séparée par une virgule. Exemple :

```
monitorExitClasses=testExits.TestExit1,testExits.testExit2
```

L'interface de l'exit utilisateur du moniteur est la suivante:

```
package com.ibm.wmqfte.exitroutine.api;

import java.util.Map;

/**
 * An interface that is implemented by classes that want to be invoked as part of
 * user exit routine processing. This interface defines a method that will be
 * invoked immediately prior to starting a task as the result of a monitor trigger
 */
public interface MonitorExit {

    /**
     * Invoked immediately prior to starting a task as the result of a monitor
     * trigger.
     *
     * @param environmentMetaData
     *     meta data about the environment in which the implementation
     *     of this method is running. This information can only be read,
     *     it cannot be updated by the implementation. The constant
     *     defined in EnvironmentMetaDataConstants class can
     *     be used to access the data held by this map.
     *
     * @param monitorMetaData
     *     meta data to associate with the monitor. The meta data passed
     *     to this method can be altered, and the changes will be
     *     reflected in subsequent exit routine invocations. This map
     *     also contains keys with IBM reserved names. These entries are
     *     defined in the MonitorMetaDataConstants class and
     *     have special semantics. The the values of the IBM reserved names
     *     cannot be modified by the exit
     *
     * @param taskDetails
     *     An XML String representing the task to be executed as a result of
     *     the monitor triggering. This XML string may be modified by the
     *     exit
     *
     * @return
     *     a monitor exit result object which is used to determine if the
     *     task should proceed, or be cancelled.
     */
    MonitorExitResult onMonitor(Map<String, String> environmentMetaData,
                               Map<String, String> monitorMetaData,
                               Reference<String> taskDetails);
}
```

Les constantes des valeurs réservées à IBM dans les métadonnées du moniteur sont les suivantes:

```
package com.ibm.wmqfte.exitroutine.api;

/**
 * Constants for IBM reserved values placed into the monitor meta data
 * maps used by the monitor exit routines.
 */
public interface MonitorMetaDataConstants {

    /**
     * The value associated with this key is the name of the trigger
     * file associated with the monitor. Any modification performed
     * to this property by user exit routines will be ignored.
     */
    final String FILE_NAME_KEY = "FILENAME";

    /**
     * The value associated with this key is the path to the trigger
     */
}
```

```

* file associated with the monitor. Any modification performed
* to this property by user exit routines will be ignored.
*/
final String FILE_PATH_KEY = "FILEPATH";

/**
* The value associated with this key is the size of the trigger
* file associated with the monitor. This will not be present in
* the cases where the size cannot be determined. Any modification
* performed to this property by user exit routines will be ignored.
*/
final String FILE_SIZE_KEY = "FILESIZE";

/**
* The value associated with this key is the local date on which
* the trigger file associated with the monitor was last modified.
* Any modification performed to this property by user exit routines
* will be ignored.
*/
final String LAST_MODIFIED_DATE_KEY = "LASTMODIFIEDDATE";

/**
* The value associated with this key is the local time at which
* the trigger file associated with the monitor was last modified.
* Any modification performed to this property by user exit routines
* will be ignored.
*/
final String LAST_MODIFIED_TIME_KEY = "LASTMODIFIEDTIME";

/**
* The value associated with this key is the UTC date on which
* the trigger file associated with the monitor was last modified.
* Any modification performed to this property by user exit routines
* will be ignored.
*/
final String LAST_MODIFIED_DATE_KEY_UTC = "LASTMODIFIEDDATEUTC";

/**
* The value associated with this key is the UTC time at which
* the trigger file associated with the monitor was last modified.
* Any modification performed to this property by user exit routines
* will be ignored.
*/
final String LAST_MODIFIED_TIME_KEY_UTC = "LASTMODIFIEDTIMEUTC";

/**
* The value associated with this key is the name of the agent on which
* the monitor is running. Any modification performed to this property by
* user exit routines will be ignored.
*/
final String MONITOR_AGENT_KEY = "AGENTNAME";
}

```

## Exemple d'exit utilisateur de moniteur

Cet exemple de classe implémente l'interface `MonitorExit`. Cet exemple ajoute une variable de substitution personnalisée dans les métadonnées du moniteur appelées `REDIRECTEDAGENT` qui sera renseignée avec la valeur `LONDON` si l'heure de la journée est impaire et la valeur `PARIS` pour les heures paires. Le code de résultat de l'exit du moniteur est défini pour toujours renvoyer `proceed`.

```

package com.ibm.wmqfte.monitor;

import java.util.Calendar;
import java.util.Map;

import com.ibm.wmqfte.exitroutine.api.MonitorExit;
import com.ibm.wmqfte.exitroutine.api.MonitorExitResult;
import com.ibm.wmqfte.exitroutine.api.Reference;

/**
* Example resource monitor user exit that changes the monitor mutable
* metadata value between 'LONDON' and 'PARIS' depending on the hour of the day.
*
*/
public class TestMonitorExit implements MonitorExit {

```

```

// custom variable that will substitute destination agent
final static String REDIRECTED_AGENT = "REDIRECTEDAGENT";

public MonitorExitResult onMonitor(
Map<String, String> environmentMetaData,
Map<String, String> monitorMetaData,
Reference<String> taskDetails) {

    // always succeed
    final MonitorExitResult result = MonitorExitResult.PROCEED_RESULT;

    final int hour = Calendar.getInstance().get(Calendar.HOUR_OF_DAY);

    if (hour%2 == 1) {
        monitorMetaData.put(REDIRECTED_AGENT, "LONDON");
    } else {
        monitorMetaData.put(REDIRECTED_AGENT, "PARIS");
    }

    return result;
}
}

```

La tâche correspondante pour un moniteur qui utilise la variable de substitution *REDIRECTEDAGENT* peut se présenter comme suit:

```

<?xml version="1.0" encoding="UTF-8"?>
<request version="4.00"
  xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="FileTransfer.xsd">
  <managedTransfer>
    <originator>
      <hostName>reportserver.com</hostName>
      <userID>USER1</userID>
    </originator>
    <sourceAgent agent="AGENT1"
      QMgr="QM1"/>
    <destinationAgent agent="{REDIRECTEDAGENT}"
      QMgr="QM2"/>
    <transferSet>
      <item mode="binary" checksumMethod="MD5">
        <source recursive="false" disposition="delete">
          <file>c:\sourcefiles\reports.doc</file>
        </source>
        <destination type="file" exist="overwrite">
          <file>c:\destinationfiles\reports.doc</file>
        </destination>
      </item>
    </transferSet>
  </managedTransfer>
</request>

```

Avant le début de ce transfert, la valeur de l'attribut agent de l'élément <destinationAgent> est remplacée par LONDON ou PARIS.

Vous devez spécifier la variable de substitution dans la classe d'exit du moniteur et dans le fichier XML de définition de tâche en majuscules.

### Concepts associés

[Personnalisation de MFT avec des exits utilisateur](#)

«Métadonnées pour les exits utilisateur MFT», à la page 2221

Il existe trois types de métadonnées qui peuvent être fournis aux routines d'exit utilisateur pour Managed File Transfer: métadonnées d'environnement, de transfert et de fichier. Ces métadonnées sont présentées sous forme de mappes de paires clé-valeur Java .

«Interfaces Java pour les exits utilisateur MFT», à la page 2231

Utilisez les rubriques de cette section pour obtenir des informations de référence sur les interfaces Java pour les routines d'exit utilisateur.

[Exits utilisateur source et cible MFT](#)

### Référence associée

«Propriétés de l'agent MFT pour les exits utilisateur», à la page 2229

Outre les propriétés standard du fichier `agent.properties`, il existe plusieurs propriétés avancées spécifiques aux routines d'exit utilisateur. Ces propriétés ne sont pas incluses par défaut. Par conséquent, si vous souhaitez les utiliser, vous devez éditer manuellement le fichier `agent.properties`. Si vous modifiez le fichier `agent.properties` alors que cet agent est en cours d'exécution, arrêtez et redémarrez l'agent pour récupérer les modifications.

## Propriétés de l'agent MFT pour les exits utilisateur

Outre les propriétés standard du fichier `agent.properties`, il existe plusieurs propriétés avancées spécifiques aux routines d'exit utilisateur. Ces propriétés ne sont pas incluses par défaut. Par conséquent, si vous souhaitez les utiliser, vous devez éditer manuellement le fichier `agent.properties`. Si vous modifiez le fichier `agent.properties` alors que cet agent est en cours d'exécution, arrêtez et redémarrez l'agent pour récupérer les modifications.

Les variables d'environnement peuvent être utilisées dans certaines propriétés Managed File Transfer qui représentent des emplacements de fichier ou de répertoire. Cela permet aux emplacements des fichiers ou des répertoires utilisés lors de l'exécution de parties du produit de varier en fonction des changements d'environnement, tels que l'utilisateur qui exécute le processus. Pour plus d'informations, voir [Variables d'environnement](#) dans les propriétés MFT.

## Propriétés de la routine d'exit utilisateur

Les routines d'exit utilisateur sont appelées dans l'ordre indiqué dans le tableau suivant. Pour plus d'informations sur le fichier `agent.properties`, voir [Propriétés avancées de l'agent: routine d'exit utilisateur](#).

Tableau 885. Propriétés d'agent pour les exits utilisateur	
Nom de la propriété	Description
<code>sourceTransferEndExitClasses</code>	Spécifie une liste de classes implémentant une routine d'exit de fin de transfert source.
<code>sourceTransferStartExitClasses</code>	Spécifie une liste de classes implémentant une routine d'exit de début de transfert source.
<code>destinationTransferStartExitClasses</code>	Spécifie une liste de classes implémentant une routine d'exit utilisateur de début de transfert de destination.
<code>destinationTransferEndExitClasses</code>	Indique une liste de classes séparées par des virgules qui implémentent une routine d'exit utilisateur de transfert de destination.
<code>exitClassPath</code>	<p>Indique une liste de répertoires, délimités par des caractères, spécifique à la plateforme, qui servent de chemins d'accès aux classes des routines d'exit utilisateur.</p> <p>Le répertoire des exits de l'agent est recherché avant toutes les entrées de ce chemin d'accès aux classes.</p> <p>Les parenthèses, les virgules (,) et les barres obliques inversées (\) sont des caractères spéciaux dans les commandes MFT et doivent être échappés avec une barre oblique inversée (\).</p> <p><b>Windows</b> Les chemins d'accès aux fichiers sous Windows peuvent être spécifiés à l'aide de doubles barres obliques inversées (\\) comme séparateur ou à l'aide de barres obliques simples (/). Par exemple:</p> <pre>exitClassPath=C:/mycomp/mqft/exits/encryptFileExit.jar; C:/mycomp/mqft/exits/fileFilter.jar.</pre> <p>La valeur de cette propriété peut contenir des variables d'environnement.</p>
<code>exitNativeLibraryPath</code>	<p>Indique une liste de répertoires, délimités par des caractères, spécifique à la plateforme, qui servent de chemins d'accès aux bibliothèques natives des routines d'exit utilisateur.</p> <p>La valeur de cette propriété peut contenir des variables d'environnement.</p>
<code>monitorExitClasses</code>	Spécifie une liste de classes implémentant une routine d'exit de moniteur, séparées par des virgules. Pour plus d'informations, voir <a href="#">Exits utilisateur du moniteur de ressources MFT</a> .

Tableau 885. Propriétés d'agent pour les exits utilisateur (suite)

Nom de la propriété	Description
protocolBridgeCredentialExitClasses	Spécifie une liste de classes implémentant une routine d'exit utilisateur des données d'identification du pont de protocole. Pour plus d'informations, voir <a href="#">Mapping credentials for a file server by using exit classes</a> .
protocolBridgePropertiesExitClasses	Spécifie une liste de classes implémentant une routine d'exit utilisateur des propriétés serveur du pont de protocole. Pour plus d'informations, voir <a href="#">ProtocolBridgePropertiesExit2: Looking up protocol file server properties</a> .
IOExitClasses	Spécifie une liste de classes implémentant une routine d'exit utilisateur d'E-S, séparées par des virgules. Ne répertorie que les classes qui implémentent l'interface IOExit et non celles qui implémentent d'autres interfaces d'exit utilisateur d'E-S, comme IOExitResourcePath et IOExitChannel. Pour plus d'informations, voir <a href="#">Using MFT transfer I/O user exits</a> .

## Ordre d'appel de l'exit

Les exits source et de destination sont appelés dans l'ordre suivant:

1. SourceTransferStartExit
2. DestinationTransferStartExit
3. DestinationTransferEndExit
4. SourceTransferEndExit

## Enchaînement des exits source et de destination

Si vous spécifiez plusieurs exits, le premier exit de la liste est appelé en premier, suivi du deuxième, et ainsi de suite. Toutes les modifications apportées par le premier exit sont transmises en tant qu'entrée à l'exit qui est ensuite appelé, etc. Par exemple, s'il existe deux exits de démarrage de transfert source, toutes les modifications apportées aux métadonnées de transfert par le premier exit sont entrées dans le deuxième exit. Chaque exit renvoie son propre résultat. Si tous les exits d'un type donné renvoient CONTINUE comme code de résultat de transfert, le résultat global est CONTINUE. Si un ou plusieurs exits renvoient CANCEL\_TRANSFER, le résultat global est CANCEL\_TRANSFER. Tous les codes de résultat et les chaînes renvoyés par les exits sont consignés dans le journal de transfert.

Si le résultat global de l'exit de démarrage de transfert source est CONTINUE, le transfert se poursuit en utilisant les modifications apportées par les exits. Si le résultat global est CANCEL\_TRANSFER, les exits de fin de transfert source sont appelés, puis le transfert est annulé. Le statut d'achèvement dans le journal de transfert est "annulé".

Si le résultat global des exits de démarrage de transfert de destination est CONTINUE, le transfert se poursuit en utilisant les modifications apportées par les exits. Si le résultat global est CANCEL\_TRANSFER, les exits de fin de transfert de destination sont appelés, puis les exits de fin de transfert source sont appelés. Enfin, le transfert est annulé. Le statut d'achèvement dans le journal de transfert est "annulé".

Si un exit source ou de destination doit transmettre des informations aux exits suivants dans la chaîne ou dans l'ordre d'exécution, il doit être effectué en mettant à jour les métadonnées de transfert. L'utilisation des métadonnées de transfert est spécifique à l'implémentation de l'exit. Par exemple, si un exit définit le résultat de retour sur CANCEL\_TRANSFER et doit communiquer aux exits suivants que le transfert a été annulé, il doit définir une valeur de métadonnées de transfert d'une manière comprise par les autres exits.

## Exemple

```
sourceTransferStartExitClasses=com.ibm.wmqfte.test.MFTTestSourceTransferStartExit
sourceTransferEndExitClasses=com.ibm.wmqfte.test.MFTTestSourceTransferEndExit
destinationTransferStartExitClasses=com.ibm.wmqfte.test.MFTTestDestinationTransferStartExit
destinationTransferEndExitClasses=com.ibm.wmqfte.test.MFTTestDestinationTransferEndExit
exitClassPath=C:/mycomp/mqft/exits/encryptFileExit.jar;C:/mycomp/mqft/exits/fileFilter.jar
```

## Concepts associés

[Personnalisation de MFT avec des exits utilisateur](#)

[«Métadonnées pour les exits utilisateur MFT», à la page 2221](#)

Il existe trois types de métadonnées qui peuvent être fournis aux routines d'exit utilisateur pour Managed File Transfer: métadonnées d'environnement, de transfert et de fichier. Ces métadonnées sont présentées sous forme de mappes de paires clé-valeur Java .

[«Interfaces Java pour les exits utilisateur MFT», à la page 2231](#)

Utilisez les rubriques de cette section pour obtenir des informations de référence sur les interfaces Java pour les routines d'exit utilisateur.

## Référence associée

[«Exits utilisateur du moniteur de ressources MFT», à la page 2225](#)

Les exits utilisateur du moniteur de ressources MFT permettent de configurer le code personnalisé à exécuter lorsque la condition de déclenchement d'un moniteur est remplie, avant le démarrage de la tâche associée.

[Variables d'environnement dans les propriétés MFT](#)

[Le fichier MFT agent.properties](#)

## Interfaces Java pour les exits utilisateur MFT

Utilisez les rubriques de cette section pour obtenir des informations de référence sur les interfaces Java pour les routines d'exit utilisateur.

### Tâches associées

[Personnalisation de MFT avec des exits utilisateur](#)

### Référence associée

[«Interface DestinationTransferStartExit.java», à la page 2235](#)

[«Interface DestinationTransferEndExit.java», à la page 2233](#)

[«Interface IOExit.java», à la page 2237](#)

[«Interface IOExitChannel.java», à la page 2239](#)

[«Interface IOExitLock.java», à la page 2241](#)

[«Interface IOExitPath.java», à la page 2241](#)

[«Interface IOExitProperties.java», à la page 2242](#)

[«Interface IOExitRecordChannel.java», à la page 2245](#)

[«Interface IOExitRecordResourcePath.java», à la page 2247](#)

[«Interface IOExitResourcePath.java», à la page 2248](#)

[«Interface IOExitWildcardPath.java», à la page 2253](#)

[«Interface MonitorExit.java», à la page 2253](#)

[«Interface ProtocolBridgeCredentialExit.java», à la page 2254](#)

[«Interface ProtocolBridgeCredentialExit2.java», à la page 2255](#)

[«Interface ProtocolBridgePropertiesExit2.java», à la page 2256](#)

[«Interface SourceTransferStartExit.java», à la page 2259](#)

[«Interface SourceTransferEndExit.java», à la page 2258](#)

## ***Interface CDCredentialExit.java***

### **CDCredentialExit.java**

```
/*  
 * Licensed Materials - Property of IBM  
 */
```

```

* "Restricted Materials of IBM"
*
* 5724-H72
*
* □ Copyright IBM Corp. 2011, 2024. All Rights Reserved.
*
* US Government Users Restricted Rights - Use, duplication or
* disclosure restricted by GSA ADP Schedule Contract with
* IBM Corp.
*/
package com.ibm.wmqfte.exitroutine.api;

import java.util.Map;

/**
 * An interface that is implemented by classes that are invoked as part of
 * user exit routine processing. This interface defines methods that are
 * invoked by a Connect:Direct bridge agent to map the IBM MQ user ID of the transfer to credentials
 * that are used to access the Connect:Direct node.
 * There will be one instance of each implementation class per Connect:Direct bridge agent. The methods
 * can be called from different threads so the methods must be synchronized.
 */
public interface CDCredentialExit {

    /**
     * Invoked once when a Connect:Direct bridge agent is started. It is intended to initialize
     * any resources that are required by the exit
     *
     * @param bridgeProperties
     *         The values of properties defined for the Connect:Direct bridge.
     *         These values can only be read, they cannot be updated by
     *         the implementation.
     *
     * @return true if the initialisation is successful and false if unsuccessful
     *         If false is returned from an exit the Connect:Direct bridge agent does not
     *         start.
     */
    public boolean initialize(final Map<String, String> bridgeProperties);

    /**
     * Invoked once per transfer to map the IBM MQ user ID in the transfer message to the
     * credentials to be used to access the Connect:Direct node.
     *
     * @param mqUserId The IBM MQ user ID from which to map to the credentials to be used
     *                 to access the Connect:Direct node
     * @param snode    The name of the Connect:Direct SNODE specified as the cdNode in the
     *                 file path. This is used to map the correct user ID and password for the
     *                 SNODE.
     * @return         A credential exit result object that contains the result of the map and
     *                 the credentials to use to access the Connect:Direct node
     */
    public CDCredentialExitResult mapMQUserId(final String mqUserId, final String snode);

    /**
     * Invoked once when a Connect:Direct bridge agent is shutdown. This method releases
     * any resources that were allocated by the exit
     *
     * @param bridgeProperties
     *         The values of properties defined for the Connect:Direct bridge.
     *         These values can only be read, they cannot be updated by
     *         the implementation.
     *
     * @return
     */
    public void shutdown(final Map<String, String> bridgeProperties); }

```

## ***Interface CredentialExitResult.java***

### **CredentialExitResult.java**

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 */

```



```

*   Copyright IBM Corp. 2008, 2024. All Rights Reserved.
*
*   US Government Users Restricted Rights - Use, duplication or
*   disclosure restricted by GSA ADP Schedule Contract with
*   IBM Corp.
*/

package com.ibm.wmqfte.exitroutine.api;

/**
 * The result of invoking a Credential mapMQUserId exit method. It is composed of a result
 * code, which determines whether the mapping of the user id was successful, and an optional
 * Credentials object if the mapping is successful.
 */
public class CredentialExitResult {

    private final CredentialExitResultCode resultCode;
    private final Credentials credentials;

    /**
     * Constructor. Creates a credential exit result object with a specified result
     * code and optionally credentials.
     *
     * @param resultCode
     *         The result code to associate with the exit result being created.
     *
     * @param credentials
     *         The credentials to associate with the exit result being created.
     *         A value of <code>null</code> can be specified to indicate no
     *         credentials. If the resultCode is USER_SUCCESSFULLY_MAPPED the
     *         credentials must be set to a non-null value,
     */
    public CredentialExitResult(CredentialExitResultCode resultCode, Credentials credentials) {
        this.resultCode = resultCode;
        this.credentials = credentials;
    }

    /**
     * Returns the result code associated with this credential exit result
     *
     * @return    the result code associated with this exit result.
     */
    public CredentialExitResultCode getResultCode() {
        return resultCode;
    }

    /**
     * Returns the credentials associated with this credential exit result
     *
     * @return    the explanation associated with this credential exit result.
     */
    public Credentials getCredentials() {
        return credentials;
    }
}

```

## Tâches associées

[Personnalisation de MFT avec des exits utilisateur](#)

## Référence associée

[«Interface SourceTransferStartExit.java», à la page 2259](#)

[«Interface DestinationTransferStartExit.java», à la page 2235](#)

[«Interface DestinationTransferEndExit.java», à la page 2233](#)

[«Interface MonitorExit.java», à la page 2253](#)

[«Interface ProtocolBridgeCredentialExit.java», à la page 2254](#)

## ***Interface DestinationTransferEndExit.java***

### **DestinationTransferEndExit.java**

```

/*
 * Licensed Materials - Property of IBM

```

```

*
* "Restricted Materials of IBM"
*
* 5724-H72
*
* □ Copyright IBM Corp. 2008, 2024. All Rights Reserved.
*
* US Government Users Restricted Rights - Use, duplication or
* disclosure restricted by GSA ADP Schedule Contract with
* IBM Corp.
*/
package com.ibm.wmqfte.exitpoint.api;

/**
 * An interface that is implemented by classes that want to be invoked as part of
 * user exit routine processing. This interface defines a method that will be
 * invoked immediately after completing a transfer on the agent acting as the
 * destination of the transfer.
 */
public interface DestinationTransferEndExit {

    /**
     * Invoked immediately after the completion of a transfer on the agent acting as
     * the destination of the transfer.
     *
     * @param transferExitResult
     *     a result object reflecting whether or not the transfer completed
     *     successfully.
     *
     * @param sourceAgentName
     *     the name of the agent acting as the source of the transfer.
     *
     * @param destinationAgentName
     *     the name of the agent acting as the destination of the
     *     transfer. This is the name of the agent that the
     *     implementation of this method will be invoked from.
     *
     * @param environmentMetaData
     *     meta data about the environment in which the implementation
     *     of this method is running. This information can only be read,
     *     it cannot be updated by the implementation. The constants
     *     defined in <code>EnvironmentMetaDataConstants</code> class can
     *     be used to access the data held by this map.
     *
     * @param transferMetaData
     *     meta data to associate with the transfer. The information can
     *     only be read, it cannot be updated by the implementation. This
     *     map may also contain keys with IBM reserved names. These
     *     entries are defined in the <code>TransferMetaDataConstants</code>
     *     class and have special semantics.
     *
     * @param fileResults
     *     a list of file transfer result objects that describe the source
     *     file name, destination file name and result of each file transfer
     *     operation attempted.
     *
     * @return
     *     an optional description to enter into the log message describing
     *     transfer completion. A value of <code>null</code> can be used
     *     when no description is required.
     */
    String onDestinationTransferEnd(TransferExitResult transferExitResult,
        String sourceAgentName,
        String destinationAgentName,
        Map<String, String>environmentMetaData,
        Map<String, String>transferMetaData,
        List<FileTransferResult>fileResults);
}

```

## Tâches associées

[Personnalisation de MFT avec des exits utilisateur](#)

## Référence associée

[«Interface SourceTransferStartExit.java», à la page 2259](#)

[«Interface SourceTransferEndExit.java», à la page 2258](#)

[«Interface DestinationTransferStartExit.java», à la page 2235](#)

[«Interface MonitorExit.java», à la page 2253](#)

## ***Interface DestinationTransferStartExit.java***

### **DestinationTransferStartExit.java**

```
/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * © Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitpoint.api;

/**
 * An interface that is implemented by classes that want to be invoked as part of
 * user exit routine processing. This interface defines a method that will be
 * invoked immediately prior to starting a transfer on the agent acting as the
 * destination of the transfer.
 */
public interface DestinationTransferStartExit {

    /**
     * Invoked immediately prior to starting a transfer on the agent acting as
     * the destination of the transfer.
     *
     * @param sourceAgentName
     *     the name of the agent acting as the source of the transfer.
     *
     * @param destinationAgentName
     *     the name of the agent acting as the destination of the
     *     transfer. This is the name of the agent that the
     *     implementation of this method will be invoked from.
     *
     * @param environmentMetaData
     *     meta data about the environment in which the implementation
     *     of this method is running. This information can only be read,
     *     it cannot be updated by the implementation. The constants
     *     defined in <code>EnvironmentMetaDataConstants</code> class can
     *     be used to access the data held by this map.
     *
     * @param transferMetaData
     *     meta data to associate with the transfer. The information can
     *     only be read, it cannot be updated by the implementation. This
     *     map may also contain keys with IBM reserved names. These
     *     entries are defined in the <code>TransferMetaDataConstants</code>
     *     class and have special semantics.
     *
     * @param fileSpecs
     *     a list of file specifications that govern the file data to
     *     transfer. The implementation of this method can modify the
     *     entries in this list and the changes will be reflected in the
     *     files transferred. However, new entries may not be added and
     *     existing entries may not be removed.
     *
     * @return
     *     a transfer exit result object which is used to determine if the
     *     transfer should proceed, or be cancelled.
     */
    TransferExitResult onDestinationTransferStart(String sourceAgentName,
        String destinationAgentName,
        Map<String, String> environmentMetaData,
        Map<String, String> transferMetaData,
        List<Reference<String>> fileSpecs);
}
```

#### **Tâches associées**

Personnalisation de MFT avec des exits utilisateur

#### **Référence associée**

«Interface SourceTransferStartExit.java», à la page 2259

[«Interface SourceTransferEndExit.java», à la page 2258](#)

[«Interface DestinationTransferEndExit.java», à la page 2233](#)

[«Interface MonitorExit.java», à la page 2253](#)

[«Interface ProtocolBridgeCredentialExit.java», à la page 2254](#)

## ***Interface FileTransferResult.java***

### **FileTransferResult.java**

```
/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * © Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */

package com.ibm.wmqfte.exitroutine.api;

/**
 * Result information about a file transfer.
 */
public interface FileTransferResult {

    /** An enumeration for the <code>getCorrelatorType()</code> method. */
    public enum CorrelationInformationType {
        /** No correlation information is available for this result */
        NONE,
        /**
         * The correlation information relates to work done in
         * IBM Sterling File Gateway.
         */
        SFG
    }

    /**
     * Returns the source file specification, from which the file was transferred.
     *
     * @return the source file specification, from which the file was
     * transferred.
     */
    String getSourceFileSpecification();

    /**
     * Returns the destination file specification, to which the file was transferred.
     *
     * @return the destination file specification, to which the file was
     * transferred. A value of <code>null</code> may be returned
     * if the transfer did not complete successfully.
     */
    String getDestinationFileSpecification();

    /**
     * Returns the result of the file transfer operation.
     *
     * @return the result of the file transfer operation.
     */
    FileExitResult getExitResult();

    /**
     * @return an enumerated value that identifies the product to which this correlating
     * information relates.
     */
    CorrelationInformationType getCorrelatorType();

    /**
     * @return the first string component of the correlating identifier that relates
     * this transfer result to work done in another product. A value of null
     * may be returned either because the other product does not utilize a

```

```

    *      string based correlation information or because there is no correlation
    *      information.
    */
String getString1Correlator();

/**
 * @return the first long component of the correlating identifier that relates
 * this transfer result to work done in another product. A value of zero
 * is returned when there is no correlation information or the other
 * product does not utilize long based correlation information or because
 * the value really is zero!
 */
long getLong1Correlator();
}

```

## Tâches associées

[Personnalisation de MFT avec des exits utilisateur](#)

## Référence associée

[«Interface SourceTransferStartExit.java», à la page 2259](#)

[«Interface DestinationTransferStartExit.java», à la page 2235](#)

[«Interface DestinationTransferEndExit.java», à la page 2233](#)

[«Interface MonitorExit.java», à la page 2253](#)

[«Interface ProtocolBridgeCredentialExit.java», à la page 2254](#)

## Interface IOExit.java

### IOExit.java

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * © Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.io.IOException;
import java.util.Map;

import com.ibm.wmqfte.exitroutine.api.IOExitRecordResourcePath.RecordFormat;

/**
 * An interface that is implemented by classes that you want to be invoked as
 * part of user exit routine processing. This interface defines methods that
 * will be invoked during transfers to perform the underlying file system I/O
 * work for WMQFTE transfers.
 * <p>
 * The {@link #initialize(Map)} method will be called once when the exit is
 * first installed. The WMQFTE agent properties are passed to this method, thus
 * enabling the exit to understand its environment.
 * <p>
 * The {@link #isSupported(String)} method will be invoked during WMQFTE
 * transfers to determine whether the user exit should be used. If the
 * {@link #isSupported(String)} method returns a value of {@code true}, the
 * {@link #newPath(String)} method will be invoked for the paths specified for
 * the transfer request. The returned {@link IOExitPath} instance from a
 * {@link #newPath(String)} method invocation will then be used by the WMQFTE
 * transfer to obtain information about the resource and to transfer data to or
 * from the resource.
 * <p>
 * To obtain transfer context for an I/O exit, a {@link SourceTransferStartExit}
 * or {@link DestinationTransferStartExit} as appropriate, should be installed
 * to enable information to be seen by this exit. The
 * {@link SourceTransferStartExit} or {@link DestinationTransferStartExit} are
 * passed the transfer's environment, metadata, and a list of file

```

```

* specifications for the transfer. The paths for the file specifications are
* the paths passed to the I/O exit's {@link #newPath(String)} method.
* <p>
* Note also that the {@link #isSupported(String)} and {@link #newPath(String)}
* methods might be called at other times by a WMQFTE agent and not just during
* transfers. For example, at transfer setup time the I/O system is queried to
* resolve the full resource paths for transfer.
*/
public interface IOExit {

    /**
     * Invoked once when the I/O exit is first required for use. It is intended
     * to initialize any resources that are required by the exit.
     *
     * @param agentProperties
     *     The values of properties defined for the WMQFTE agent. These
     *     values can only be read, they cannot be updated by the
     *     implementation.
     * @return {@code true} if the initialization is successful and {@code
     *         false} if unsuccessful. If {@code false} is returned from an
     *         exit, the exit will not be used.
     */
    boolean initialize(final Map<String, String> agentProperties);

    /**
     * Indicates whether this I/O user exit supports the specified path.
     * <p>
     * This method is used by WMQFTE to determine whether the I/O user exit
     * should be used within a transfer. If no I/O user exit returns true for
     * this method, the default WMQFTE file I/O function will be used.
     *
     * @param path
     *     The path to the required I/O resource.
     * @return {@code true} if the specified path is supported by the I/O exit,
     *         {@code false} otherwise
     */
    boolean isSupported(String path);

    /**
     * Obtains a new {@link IOExitPath} instance for the specified I/O resource
     * path.
     * <p>
     * This method will be invoked by WMQFTE only if the
     * {@link #isSupported(String)} method has been called for the path and
     * returned {@code true}.
     *
     * @param path
     *     The path to the required I/O resource.
     * @return A {@link IOExitPath} instance for the specified path.
     * @throws IOException
     *     If the path cannot be created for any reason.
     */
    IOExitPath newPath(String path) throws IOException;

    /**
     * Obtains a new {@link IOExitPath} instance for the specified I/O resource
     * path and passes record format and length information required by the
     * WMQFTE transfer.
     * <p>
     * Typically this method will be called for the following cases:
     * <ul>
     * <li>A path where a call to {@link #newPath(String)} has previously
     * returned a {@link IOExitRecordResourcePath} instance and WMQFTE is
     * re-establishing a new {@link IOExitPath} instance for the path, from an
     * internally-serialized state. The passed recordFormat and recordLength
     * will be the same as those for the original
     * {@link IOExitRecordResourcePath} instance.</li>
     * <li>A transfer destination path where the source of the transfer is
     * record oriented. The passed recordFormat and recordLength will be the
     * same as those for the source.</li>
     * </ul>
     * The implementation can act on the record format and length information as
     * deemed appropriate. For example, for a destination agent if the
     * destination does not already exist and the source of the transfer is
     * record oriented, the passed recordFormat and recordLength information
     * could be used to create an appropriate record-oriented destination path.
     * If the destination path already exists, the passed recordFormat and
     * recordLength information could be used to perform a compatibility check
     * and throw an {@link IOException} if the path is not compatible. A
     * compatibility check could ensure that a record oriented path's record
     * format is the same as the passed record format or that the record length
     * is greater or equal to the passed record length.
     */

```

```

* <p>
* This method will be invoked by WMQFTE only if the
* {@link #isSupported(String)} method has been called for the path and
* returned {@code true}.
*
* @param path
*     The path to the required I/O resource.
* @param recordFormat
*     The advised record format.
* @param recordLength
*     The advised record length.
* @return A {@link IOExitPath} instance for the specified path.
* @throws IOException
*     If the path cannot be created for any reason. For example,
*     the passed record format or length is incompatible with the
*     path's actual record format or length.
*/
IOExitPath newPath(String path, RecordFormat recordFormat, int recordLength)
    throws IOException;

```

### Tâches associées

[Utilisation des exits utilisateur d'E-S de transfert MFT](#)

[Personnalisation de MFT avec des exits utilisateur](#)

## Interface *IOExitChannel.java*

### IOExitChannel.java

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * © Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.io.IOException;
import java.nio.ByteBuffer;

/**
 * Represents a channel that enables data to be read from or written to an
 * {@link IOExitResourcePath} resource.
 */
public interface IOExitChannel {

    /**
     * Obtains the data size for the associated {@link IOExitResourcePath} in
     * bytes.
     *
     * @return The data size in bytes.
     * @throws IOException
     *     If a problem occurs while attempting obtain the size.
     */
    long size() throws IOException;

    /**
     * Closes the channel, flushing any buffered write data to the resource and
     * releasing any locks.
     *
     * @throws RecoverableIOException
     *     If a recoverable problem occurs while closing the resource.
     *     This means that WMQFTE can attempt to recover the transfer.
     * @throws IOException
     *     If some other I/O problem occurs. For example, the channel might
     *     already be closed.
     */
    void close() throws RecoverableIOException, IOException;

}

```

```

* Reads data from this channel into the given buffer, starting at this
* channel's current position, and updates the current position by the
* amount of data read.
* <p>
* Data is copied into the buffer starting at its current position and up to
* its limit. On return, the buffer's position is updated to reflect the
* number of bytes read.
*
* @param buffer
*       The buffer that the data is to be copied into.
* @return The number of bytes read, which might be zero, or -1 if the end of
*         data has been reached.
* @throws RecoverableIOException
*         If a recoverable problem occurs while reading the data. For a
*         WMQFTE transfer this means that it will attempt to recover.
* @throws IOException
*         If some other I/O problem occurs. For a WMQFTE transfer this
*         means that it will be failed.
*/
int read(ByteBuffer buffer) throws RecoverableIOException, IOException;

/**
* Writes data to this channel from the given buffer, starting at this
* channel's current position, and updates the current position by the
* amount of data written. The channel's resource is grown to accommodate
* the data, if necessary.
* <p>
* Data is copied from the buffer starting at its current position and up to
* its limit. On return, the buffer's position is updated to reflect the
* number of bytes written.
*
* @param buffer
*       The buffer containing the data to be written.
* @return The number of bytes written, which might be zero.
* @throws RecoverableIOException
*         If a recoverable problem occurs while writing the data. For a
*         WMQFTE transfer this means that it will attempt to recover.
* @throws IOException
*         If some other I/O problem occurs. For a WMQFTE transfer this
*         means that it will be failed.
*/
int write(ByteBuffer buffer) throws RecoverableIOException, IOException;

/**
* Forces any updates to this channel's resource to be written to its
* storage device.
* <p>
* This method is required to force changes to both the resource's content
* and any associated metadata to be written to storage.
*
* @throws RecoverableIOException
*         If a recoverable problem occurs while performing the force.
*         For a WMQFTE transfer this means that it will attempt to
*         recover.
* @throws IOException
*         If some other I/O problem occurs. For a WMQFTE transfer this
*         means that it will be failed.
*/
void force() throws RecoverableIOException, IOException;

/**
* Attempts to lock the entire resource associated with the channel for
* shared or exclusive access.
* <p>
* The intention is for this method not to block if the lock is currently
* unavailable.
*
* @param shared
*       {@code true} if a shared lock is required, {@code false} if an
*       exclusive lock is required.
* @return A {@link IOExitLock} instance representing the newly acquired
*         lock or null if the lock cannot be obtained.
* @throws IOException
*         If a problem occurs while attempting to acquire the lock.
*/
IOExitLock tryLock(boolean shared) throws IOException;
}

```

## Tâches associées

Utilisation des exits utilisateur d'E-S de transfert MFT



## ***Interface IOExitLock.java***

### **IOExitLock.java**

```
/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * © Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.io.IOException;

/**
 * Represents a lock on a resource for either shared or exclusive access.
 * {@link IOExitLock} instances are returned from
 * {@link IOExitChannel#tryLock(boolean)} calls and WMQFTE will request the
 * release of the lock at the appropriate time during a transfer. Additionally, when
 * a {@link IOExitChannel#close()} method is called it will be the
 * responsibility of the channel to release any associated locks.
 */
public interface IOExitLock {

    /**
     * Releases the lock.
     * <p>
     * After this method has been successfully called the lock is to be deemed as invalid.
     *
     * @throws IOException
     *         If the channel associated with the lock is not open or
     *         another problem occurs while attempting to release the lock.
     */
    void release() throws IOException;

    /**
     * Indicates whether this lock is valid.
     * <p>
     * A lock is considered valid until its @ {@link #release()} method is
     * called or the associated {@link IOExitChannel} is closed.
     *
     * @return {@code true} if this lock is valid, {@code false} otherwise.
     */
    boolean isValid();

    /**
     * @return {@code true} if this lock is for shared access, {@code false} if
     *         this lock is for exclusive access.
     */
    boolean isShared();
}
```

### **Tâches associées**

Utilisation des exits utilisateur d'E-S de transfert MFT

[Personnalisation de MFT avec des exits utilisateur](#)

## ***Interface IOExitPath.java***

### **IOExitPath.java**

```
/*
 * Licensed Materials - Property of IBM
 *
```

```

* "Restricted Materials of IBM"
*
* 5724-H72
*
* □ Copyright IBM Corp. 2011, 2024. All Rights Reserved.
*
* US Government Users Restricted Rights - Use, duplication or
* disclosure restricted by GSA ADP Schedule Contract with
* IBM Corp.
*/
package com.ibm.wmqfte.exitroutine.api;

/**
 * Represents an abstract path that can be inspected and queried by WMQFTE for
 * transfer purposes.
 * <p>
 * There are two types of path supported:
 * <ul>
 * <li>{@link IOExitResourcePath} - Represents a path that denotes a data
 * resource. For example, a file, directory, or group of database records.</li>
 * <li>{@link IOExitWildcardPath} - Represents a wildcard path that can be
 * expanded to multiple {@link IOExitResourcePath} instances.</li>
 * </ul>
 */
public abstract interface IOExitPath {

    /**
     * Obtains the abstract path as a {@link String}.
     *
     * @return The abstract path as a {@link String}.
     */
    String getPath();

    /**
     * Obtains the name portion of this abstract path as a {@link String}.
     * <p>
     * For example, a UNIX-style file system implementation evaluates the
     * path {@code /home/fteuser/file1.txt} as having a name of {@code
     * file1.txt}.
     *
     * @return the name portion of this abstract path as a {@link String}.
     */
    String getName();

    /**
     * Obtains the parent path for this abstract path as a {@link String}.
     * <p>
     * For example, a UNIX-style file system implementation evaluates the
     * path {@code /home/fteuser/file1.txt} as having a parent path of {@code
     * /home/fteuser}.
     *
     * @return The parent portion of the path as a {@link String}.
     */
    String getParent();

    /**
     * Obtains the abstract paths that match this abstract path.
     * <p>
     * If this abstract path denotes a directory resource, a list of paths
     * for all resources within the directory are returned.
     * <p>
     * If this abstract path denotes a wildcard, a list of all paths
     * matching the wildcard are returned.
     * <p>
     * Otherwise null is returned, because this abstract path probably denotes a
     * single file resource.
     *
     * @return An array of {@link IOExitResourcePath}s that
     * match this path, or null if this method is not applicable.
     */
    IOExitResourcePath[] listPaths();
}

```

## Tâches associées

[Utilisation des exits utilisateur d'E-S de transfert MFT](#)

[Personnalisation de MFT avec des exits utilisateur](#)

## ***Interface IOExitProperties.java***

## IOExitProperties.java

```
/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * © Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

/**
 * Properties that determine how WMQFTE treats an {@link IOExitPath} for certain
 * aspects of I/O. For example, whether to use intermediate files.
 */
public class IOExitProperties {

    private boolean rereadSourceOnRestart = true;
    private boolean rechecksumSourceOnRestart = true;
    private boolean rechecksumDestinationOnRestart = true;
    private boolean useIntermediateFileAtDestination = true;
    private boolean requiresSingleThreadedChannelIO = false;

    /**
     * Determines whether the I/O exit implementation expects the resource to be
     * re-read from the start if a transfer is restarted.
     *
     * @return {@code true} if, on restart, the I/O exit expects the source
     * resource to be opened at the beginning and re-read from the
     * beginning (the {@link IOExitPath#openForRead(long)} method is
     * always invoked with 0L as an argument). {@code false} if, on
     * restart, the I/O exit expects the source to be opened at the
     * offset that the source agent intends to start reading from (the
     * {@link IOExitPath#openForRead(long)} method can be invoked with a
     * non-zero value as its argument).
     */
    public boolean getRereadSourceOnRestart() {
        return rereadSourceOnRestart;
    }

    /**
     * Sets the value to determine whether the I/O exit implementation expects
     * the resource to be re-read from the beginning if a transfer is restarted.
     * <p>
     * The default is {@code true}. The I/O exit should call this method when
     * required to change this value.
     *
     * @param rereadSourceOnRestart
     *        {@code true} if, on restart, the I/O exit expects the source
     * resource to be opened at the beginning and re-read from the
     * beginning (the {@link IOExitPath#openForRead(long)} method
     * is always invoked with 0L as an argument). {@code false}
     * if, on restart, the I/O exit expects the source to be opened
     * at the offset that the source agent intends to start reading
     * from (the {@link IOExitPath#openForRead(long)} method can be
     * invoked with a non-zero value as its argument).
     */
    public void setRereadSourceOnRestart(boolean rereadSourceOnRestart) {
        this.rereadSourceOnRestart = rereadSourceOnRestart;
    }

    /**
     * Determines whether the I/O exit implementation requires the source
     * resource to be re-checksummed if the transfer is restarted.
     * Re-checksumming takes place only if the
     * {@link #getRereadSourceOnRestart()} method returns {@code true}.
     *
     * @return {@code true} if, on restart, the I/O exit expects the already-
     * transferred portion of the source to be re-checksummed for
     * inconsistencies. Use this option in environments
     * where the source could be changed during a restart. {@code
     * false} if, on restart, the I/O exit does not require the
     * already-transferred portion of the source to be re-checksummed.
     */
}
```

```

public boolean getRechecksumSourceOnRestart() {
    return rechecksumSourceOnRestart;
}

/**
 * Sets the value to determine whether the I/O exit implementation requires
 * the source resource to be re-checksummed if the transfer is restarted.
 * Re-checksumming takes place only if the
 * {@link #getRereadSourceOnRestart()} method returns {@code true}.
 * <p>
 * The default is {@code true}. The I/O exit should call this method when
 * required to change this value.
 *
 * @param rechecksumSourceOnRestart
 *     {@code true} if, on restart, the I/O exit expects the already
 *     transferred portion of the source to be re-checksummed
 *     for inconsistencies. Use this option in environments
 *     where the source could be changed during a restart.
 *     {@code false} if, on restart, the I/O exit does not
 *     require the already-transferred portion of the source to be
 *     re-checksummed.
 */
public void setRechecksumSourceOnRestart(boolean rechecksumSourceOnRestart) {
    this.rechecksumSourceOnRestart = rechecksumSourceOnRestart;
}

/**
 * Determines whether the I/O exit implementation requires the destination
 * resource to be re-checksummed if the transfer is restarted.
 *
 * @return {@code true} if, on restart, the I/O exit expects the already
 *     transferred portion of the destination to be re-checksummed to
 *     check for inconsistencies. This option should be used in
 *     environments where the destination could have been changed while
 *     a restart is occurring. {@code false} if, on restart, the I/O exit
 *     does not require the already transferred portion of the
 *     destination to be re-checksummed.
 */
public boolean getRechecksumDestinationOnRestart() {
    return rechecksumDestinationOnRestart;
}

/**
 * Sets the value to determine whether the I/O exit implementation requires
 * the destination resource to be re-checksummed if the transfer is
 * restarted.
 * <p>
 * The default is {@code true}. The I/O exit should call this method when
 * required to change this value.
 *
 * @param rechecksumDestinationOnRestart
 *     {@code true} if, on restart, the I/O exit expects the already-
 *     transferred portion of the destination to be re-checksummed
 *     for inconsistencies. Use this option in environments
 *     where the destination could have been changed during a
 *     restart. {@code false} if, on restart, the I/O exit does not
 *     require the already-transferred portion of the destination
 *     to be re-checksummed.
 */
public void setRechecksumDestinationOnRestart(
    boolean rechecksumDestinationOnRestart) {
    this.rechecksumDestinationOnRestart = rechecksumDestinationOnRestart;
}

/**
 * Determines whether the I/O exit implementation requires the use of an
 * intermediate file when writing the data at the destination. The
 * intermediate file mechanism is typically used to prevent an incomplete
 * destination resource from being processed.
 *
 * @return {@code true} if data should be written to an intermediate file at
 *     the destination and then renamed (to the requested destination
 *     path name as specified in the transfer request) after the transfer is
 *     complete. {@code false} if data should be written directly to the
 *     requested destination path name without the use of an
 *     intermediate file.
 */
public boolean getUseIntermediateFileAtDestination() {
    return useIntermediateFileAtDestination;
}

```

```

* Sets the value to determine whether the I/O exit implementation requires
* the use of an intermediate file when writing the data at the destination.
* The intermediate file mechanism is typically used to prevent an
* incomplete destination resource from being processed.
*
* <p>
* The default is {@code true}. The I/O exit should call this method when
* required to change this value.
*
* @param useIntermediateFileAtDestination
*     {@code true} if data should be written to an intermediate file
*     at the destination and then renamed (to the requested
*     destination path name as specified in the transfer request) after
*     the transfer is complete. {@code false} if data should be written
*     directly to the requested destination path name without the
*     use of an intermediate file
*/
public void setUseIntermediateFileAtDestination(
    boolean useIntermediateFileAtDestination) {
    this.useIntermediateFileAtDestination = useIntermediateFileAtDestination;
}

/**
 * Determines whether the I/O exit implementation requires
 * {@link IOExitChannel} instances to be accessed by a single thread only.
 *
 * @return {@code true} if {@link IOExitChannel} instances are to be
 *     accessed by a single thread only.
 */
public boolean requiresSingleThreadedChannelIO() {
    return requiresSingleThreadedChannelIO;
}

/**
 * Sets the value to determine whether the I/O exit implementation requires
 * channel operations for a particular instance to be accessed by a
 * single thread only.
 *
 * <p>
 * For certain I/O implementations it is necessary that resource path
 * operations such as open, read, write, and close are invoked only from a
 * single execution {@link Thread}. When set {@code true}, WMQFTE ensures
 * that the following are invoked on a single thread:
 *
 * <ul>
 * <li>{@link IOExitResourcePath#openForRead(long)} method and all methods of
 * the returned {@link IOExitChannel} instance.</li>
 * <li>{@link IOExitResourcePath#openForWrite(boolean)} method and all
 * methods of the returned {@link IOExitChannel} instance.</li>
 * </ul>
 *
 * <p>
 * This has a slight performance impact, hence enable single-threaded channel
 * I/O only when absolutely necessary.
 *
 * <p>
 * The default is {@code false}. The I/O exit should call this method when
 * required to change this value.
 *
 * @param requiresSingleThreadedChannelIO
 *     {@code true} if {@link IOExitChannel} instances are to be
 *     accessed by a single thread only.
 */
public void setRequiresSingleThreadedChannelIO(boolean requiresSingleThreadedChannelIO) {
    this.requiresSingleThreadedChannelIO = requiresSingleThreadedChannelIO;
}
}
}

```

### Tâches associées

Utilisation des exits utilisateur d'E-S de transfert MFT  
 Personnalisation de MFT avec des exits utilisateur

### Interface *IOExitRecordChannel.java*

#### IOExitRecordChannel.java

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"

```

```

*
* 5724-H72
*
*   Copyright IBM Corp. 2011, 2024. All Rights Reserved.
*
* US Government Users Restricted Rights - Use, duplication or
* disclosure restricted by GSA ADP Schedule Contract with
* IBM Corp.
*/
package com.ibm.wmqfte.exitroutine.api;

import java.io.IOException;
import java.nio.ByteBuffer;

/**
 * Represents a channel that enables records of data to be read from or written
 * to an {@link IOExitRecordResourcePath} resource.
 * <p>
 * This is an extension of the {@link IOExitChannel} interface such that the
 * {@link #read(java.nio.ByteBuffer)} and {@link #write(java.nio.ByteBuffer)}
 * methods are expected to deal in whole records of data only. That is, the
 * {@link java.nio.ByteBuffer} returned from the read method and passed to the
 * write method is assumed to contain one or more complete records.
 */
public interface IOExitRecordChannel extends IOExitChannel {

    /**
     * Reads records from this channel into the given buffer, starting at this
     * channel's current position, and updates the current position by the
     * amount of data read.
     * <p>
     * Record data is copied into the buffer starting at its current position
     * and up to its limit. On return, the buffer's position is updated to
     * reflect the number of bytes read.
     * <p>
     * Only whole records are copied into the buffer.
     * <p>
     * For a fixed-record-format resource, this might be multiple records. The
     * amount of data in the return buffer does not necessarily need to be a
     * multiple of the record length, but the last record is still to be treated
     * as a complete record and padded as required by the caller.
     * <p>
     * For a variable-format resource, this is a single whole record of a size
     * corresponding to the amount of return data or multiple whole records with
     * all except the last being treated as records of maximum size.
     *
     * @param buffer
     *         The buffer that the record data is to be copied into.
     * @return The number of bytes read, which might be zero, or -1 if the end of
     *         data has been reached.
     * @throws RecoverableIOException
     *         If a recoverable problem occurs while reading the data. For a
     *         WMQFTE transfer this means that it will attempt to recover.
     * @throws IOException
     *         If some other I/O problem occurs, for example, if the passed
     *         buffer is insufficient to contain at least one complete
     *         record). For a WMQFTE transfer this means that it will be
     *         failed.
     */
    int read(ByteBuffer buffer) throws RecoverableIOException, IOException;

    /**
     * Writes records to this channel from the given buffer, starting at this
     * channel's current position, and updates the current position by the
     * amount of data written. The channel's resource is grown to accommodate
     * the data, if necessary.
     * <p>
     * Record data is copied from the buffer starting at its current position
     * and up to its limit. On return, the buffer's position is updated to
     * reflect the number of bytes written.
     * <p>
     * The buffer is expected to contain only whole records.
     * <p>
     * For a fixed-record-format resource, this might be multiple records and if
     * there is insufficient data in the buffer for a complete record, the
     * record is to be padded as required to complete the record.
     * <p>
     * For a variable-record format resource the buffer is normally expected to
     * contain a single record of length corresponding to the amount of data
     * within the buffer. However, if the amount of data within the buffer
     * exceeds the maximum record length, the implementation can either:
     * <ol>

```

```

* <li>throw an {@link IOException} indicating that it cannot handle the
* situation.</li>
* <li>Consume a record's worth of data from the buffer, leaving the remaining
* data within the buffer.</li>
* <li>Consume all the buffer data and just write what it can to the current
* record. This effectively truncates the data.</li>
* <li>Consume all the buffer data and write to multiple records.</li>
* </ol>
*
* @param buffer
*         The buffer containing the data to be written.
* @return The number of bytes written, which might be zero.
* @throws RecoverableIOException
*         If a recoverable problem occurs while writing the data. For a
*         WMQFTE transfer this means that it will attempt to recover.
* @throws IOException
*         If some other I/O problem occurs. For a WMQFTE transfer this
*         means that it will be failed.
*/
int write(ByteBuffer buffer) throws RecoverableIOException, IOException;
}

```

### Tâches associées

[Utilisation des exits utilisateur d'E-S de transfert MFT](#)

[Personnalisation de MFT avec des exits utilisateur](#)

## **Interface IOExitRecordResourcePath.java**

### IOExitRecordResourcePath.java

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * © Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.io.IOException;

/**
 * Represents a path that denotes a record-oriented data resource (for example,
 * a z/OS data set). It allows the data to be located, the record format to be
 * understood, and {@link IOExitRecordChannel} instances to be created for read
 * or write operations.
 */
public interface IOExitRecordResourcePath extends IOExitResourcePath {

    /**
     * Record formats for record-oriented resources.
     */
    public enum RecordFormat {
        FIXED, VARIABLE
    }

    /**
     * Obtains the record length for records that are maintained by the resource
     * denoted by this abstract path.
     * <p>
     * For a resource with fixed-length records, the data for each record read
     * and written is assumed to be this length.
     * <p>
     * For a resource with variable-length records, this is the maximum length
     * for a record's data.
     * <p>
     * This method should return a value greater than zero, otherwise it can
     * result in the failure of a WMQFTE transfer that involves this abstract
     * path.
     */
}

```

```

*
* @return The record length, in bytes, for records maintained by the
*         resource.
*/
int getRecordLength();

/**
 * Obtains record format, as a {@link RecordFormat} instance, for records
 * that are maintained by the resource denoted by this abstract path.
 *
 * @return A {@link RecordFormat} instance for the record format for records
 *         that are maintained by the resource denoted by this abstract
 *         path.
 */
RecordFormat getRecordFormat();

/**
 * Opens a {@link IOExitRecordChannel} instance for reading data from the
 * resource denoted by this abstract path. The current data byte position
 * for the resource is expected to be the passed position value, such that
 * when {@link IOExitRecordChannel#read(java.nio.ByteBuffer)} is called,
 * data starting from that position is read.
 * <p>
 * Note that the data byte read position will be on a record boundary.
 *
 * @param position
 *         The required data byte read position.
 * @return A new {@link IOExitRecordChannel} instance allowing data to be
 *         read from the resource denoted by this abstract path.
 * @throws RecoverableIOException
 *         If a recoverable problem occurs while attempting to open the
 *         resource for reading. This means that WMQFTE can attempt to
 *         recover the transfer.
 * @throws IOException
 *         If some other I/O problem occurs.
 */
IOExitRecordChannel openForRead(long position)
    throws RecoverableIOException, IOException;

/**
 * Opens a {@link IOExitRecordChannel} instance for writing data to the
 * resource denoted by this abstract path. Writing of data, using the
 * {@link IOExitRecordChannel#write(java.nio.ByteBuffer)} method, starts at
 * either the beginning of the resource or end of the current data for the
 * resource, depending on the specified append parameter.
 *
 * @param append
 *         When {@code true} indicates that data written to the resource
 *         should be appended to the end of the current data. When
 *         {@code false} indicates that writing of data is to start at
 *         the beginning of the resource; any existing data is lost.
 * @return A new {@link IOExitRecordChannel} instance allowing data to be
 *         written to the resource denoted by this abstract path.
 * @throws RecoverableIOException
 *         If a recoverable problem occurs while attempting to open the
 *         resource for writing. This means that WMQFTE can attempt to
 *         recover the transfer.
 * @throws IOException
 *         If some other I/O problem occurs.
 */
IOExitRecordChannel openForWrite(boolean append)
    throws RecoverableIOException, IOException;
}

```

### Tâches associées

Utilisation des exits utilisateur d'E-S de transfert MFT  
 Personnalisation de MFT avec des exits utilisateur

### Interface *IOExitResourcePath.java*

#### IOExitResourcePath.java

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"

```



```

*
* 5724-H72
*
*   Copyright IBM Corp. 2011, 2024. All Rights Reserved.
*
* US Government Users Restricted Rights - Use, duplication or
* disclosure restricted by GSA ADP Schedule Contract with
* IBM Corp.
*/
package com.ibm.wmqfte.exitroutine.api;

import java.io.IOException;

/**
 * Represents a path that denotes a data resource (for example, a file,
 * directory, or group of database records). It allows the data to be located
 * and {@link IOExitChannel} instances to be created for read or write
 * operations.
 * <p>
 * There are two types of data resources as follows:
 * <ul>
 * <li>Directory - a container for other data resources. The
 * {@link #isDirectory()} method returns {@code true} for these.</li>
 * <li>File - a data container. This allows data to be read from or written to
 * it. The {@link #isFile()} method returns {@code true} for these.</li>
 * </ul>
 */
public interface IOExitResourcePath extends IOExitPath {

    /**
     * Creates a new {@link IOExitResourcePath} instance for a child path of the
     * resource denoted by this abstract path.
     * <p>
     * For example, with a UNIX-style path, {@code
     * IOExitResourcePath("/home/fteuser/test").newPath("subtest")} could be
     * equivalent to: {@code IOExitResourcePath("/home/fteuser/test/subtest")}
     *
     * @param child
     *         The child path name.
     * @return A new {@link IOExitResourcePath} instance that represents a child
     *         of this path.
     */
    IOExitResourcePath newPath(final String child);

    /**
     * Creates the directory path for the resource denoted by this abstract
     * path, including any necessary but nonexistent parent directories. If the
     * directory path already exists, this method has no effect.
     * <p>
     * If this operation fails, it might have succeeded in creating some of the
     * necessary parent directories.
     *
     * @throws IOException
     *         If the directory path cannot be fully created, when it does
     *         not already exist.
     */
    void makePath() throws IOException;

    /**
     * Obtains the canonical path of the abstract path as a {@link String}.
     * <p>
     * A canonical path is defined as being absolute and unique. For example,
     * the path can be represented as UNIX-style relative path: {@code
     * test/file.txt} but the absolute and unique canonical path representation
     * is: {@code /home/fteuser/test/file.txt}
     *
     * @return The canonical path as a {@link String}.
     * @throws IOException
     *         If the canonical path cannot be determined for any reason.
     */
    String getCanonicalPath() throws IOException;

    /**
     * Tests if this abstract path is an absolute path.
     * <p>
     * For example, a UNIX-style path, {@code /home/fteuser/test} is an absolute
     * path, whereas {@code fteuser/test} is not.
     *
     * @return {@code true} if this abstract path is an absolute path, {@code
     *         false} otherwise.
     */
    boolean isAbsolute();

```

```

/**
 * Tests if the resource denoted by this abstract path exists.
 *
 * @return {@code true} if the resource denoted by this abstract path
 *         exists, {@code false} otherwise.
 * @throws IOException
 *         If the existence of the resource cannot be determined for any
 *         reason.
 */
boolean exists() throws IOException;

/**
 * Tests whether the calling application can read the resource denoted by
 * this abstract path.
 *
 * @return {@code true} if the resource for this path exists and can be
 *         read, {@code false} otherwise.
 * @throws IOException
 *         If a problem occurs while attempting to determine if the
 *         resource can be read.
 */
boolean canRead() throws IOException;

/**
 * Tests whether the calling application can modify the resource denoted by
 * this abstract path.
 *
 * @return {@code true} if the resource for this path exists and can be
 *         modified, {@code false} otherwise.
 * @throws IOException
 *         If a problem occurs while attempting to determine if the
 *         resource can be modified.
 */
boolean canWrite() throws IOException;

/**
 * Tests whether the specified user is permitted to read the resource
 * denoted by this abstract path.
 * <p>
 * When WMQFTE invokes this method, the user identifier is the MQMD user
 * identifier for the requesting transfer.
 *
 * @param userId
 *         User identifier to test for access.
 * @return {@code true} if the resource for this abstract path exists and is
 *         permitted to be read by the specified user, {@code false}
 *         otherwise.
 * @throws IOException
 *         If a problem occurs while attempting to determine if the user
 *         is permitted to read the resource.
 */
boolean readPermitted(String userId) throws IOException;

/**
 * Tests whether the specified user is permitted to modify the resource
 * denoted by this abstract path.
 * <p>
 * When WMQFTE invokes this method, the user identifier is the MQMD user
 * identifier for the requesting transfer.
 *
 * @param userId
 *         User identifier to test for access.
 * @return {@code true} if the resource for this abstract path exists and is
 *         permitted to be modified by the specified user, {@code false}
 *         otherwise.
 * @throws IOException
 *         If a problem occurs while attempting to determine if the user
 *         is permitted to modify the resource.
 */
boolean writePermitted(String userId) throws IOException;

/**
 * Tests if the resource denoted by this abstract path is a directory-type
 * resource.
 *
 * @return {@code true} if the resource denoted by this abstract path is a
 *         directory type resource, {@code false} otherwise.
 */
boolean isDirectory();
/**

```

```

* Creates the resource denoted by this abstract path, if it does not
* already exist.
*
* @return {@code true} if the resource does not exist and was successfully
*         created, {@code false} if the resource already existed.
* @throws RecoverableIOException
*         If a recoverable problem occurs while attempting to create
*         the resource. This means that WMQFTE can attempt to recover
*         the transfer.
* @throws IOException
*         If some other I/O problem occurs.
*/
boolean createNewPath() throws RecoverableIOException, IOException;

/**
* Tests if the resource denoted by this abstract path is a file-type
* resource.
*
* @return {@code true} if the resource denoted by this abstract path is a
*         file type resource, {@code false} otherwise.
*/
boolean isFile();

/**
* Obtains the last modified time for the resource denoted by this abstract
* path.
* <p>
* This time is measured in milliseconds since the epoch (00:00:00 GMT,
* January 1, 1970).
*
* @return The last modified time for the resource denoted by this abstract
*         path, or a value of 0L if the resource does not exist or a
*         problem occurs.
*/
long lastModified();

/**
* Deletes the resource denoted by this abstract path.
* <p>
* If the resource is a directory, it must be empty for the delete to work.
*
* @throws IOException
*         If the delete of the resource fails for any reason.
*/
void delete() throws IOException;

/**
* Renames the resource denoted by this abstract path to the specified
* destination abstract path.
* <p>
* The rename should still be successful if the resource for the specified
* destination abstract path already exists and it is possible to replace
* it.
*
* @param destination
*         The new abstract path for the resource denoted by this
*         abstract path.
* @throws IOException
*         If the rename of the resource fails for any reason.
*/
void renameTo(IOExceptionResourcePath destination) throws IOException;

/**
* Creates a new path to use for writing to a temporary resource that did
* not previously exist.
* <p>
* The implementation can choose the abstract path name for the temporary
* resource. However, for clarity and problem diagnosis, the abstract path
* name for the temporary resource should be based on this abstract path
* name with the specified suffix appended and additional characters to make
* the path unique (for example, sequence numbers), as required.
* <p>
* When WMQFTE transfers data to a destination it normally attempts to first
* write to a temporary resource then on transfer completion renames the
* temporary resource to the required destination. This method is called by
* WMQFTE to create a new temporary resource path. The returned path should
* be new and the resource should not previously exist.
*
* @param suffix
*         Recommended suffix to use for the generated temporary path.
*
* @return A new {@link IOExceptionResourcePath} instance for the temporary

```

```

*         resource path, that did not previously exist.
* @throws RecoverableIOException
*         If a recoverable problem occurs whilst attempting to create
*         the temporary resource. This means that WMQFTE can attempt to
*         recover the transfer.
* @throws IOException
*         If some other I/O problem occurs.
*/
IOExitResourcePath createTempPath(String suffix)
    throws RecoverableIOException, IOException;

/**
* Opens a {@link IOExitChannel} instance for reading data from the resource
* denoted by this abstract path. The current data byte position for the
* resource is expected to be the passed position value, such that when
* {@link IOExitChannel#read(java.nio.ByteBuffer)} is called, data starting
* from that position is read.
*
* @param position
*         The required data byte read position.
* @return A new {@link IOExitChannel} instance allowing data to be read
*         from the resource denoted by this abstract path.
* @throws RecoverableIOException
*         If a recoverable problem occurs while attempting to open the
*         resource for reading. This means that WMQFTE can attempt to
*         recover the transfer.
* @throws IOException
*         If some other I/O problem occurs.
*/
IOExitChannel openForRead(long position) throws RecoverableIOException,
    IOException;

/**
* Opens a {@link IOExitChannel} instance for writing data to the resource
* denoted by this abstract path. Writing of data, using the
* {@link IOExitChannel#write(java.nio.ByteBuffer)} method, starts at either
* the beginning of the resource or end of the current data for the
* resource, depending on the specified append parameter.
*
* @param append
*         When {@code true} indicates that data written to the resource
*         should be appended to the end of the current data. When
*         {@code false} indicates that writing of data is to start at
*         the beginning of the resource; any existing data is lost.
* @return A new {@link IOExitChannel} instance allowing data to be written
*         to the resource denoted by this abstract path.
* @throws RecoverableIOException
*         If a recoverable problem occurs whilst attempting to open the
*         resource for writing. This means that WMQFTE can attempt to
*         recover the transfer.
* @throws IOException
*         If some other I/O problem occurs.
*/
IOExitChannel openForWrite(boolean append) throws RecoverableIOException,
    IOException;

/**
* Tests if the resource denoted by this abstract path is in use by another
* application. Typically, this is because another application has a lock on
* the resource either for shared or exclusive access.
*
* @return {code true} if resource denoted by this abstract path is in use
*         by another application, {@code false} otherwise.
*/
boolean inUse();

/**
* Obtains a {@link IOExitProperties} instance for properties associated
* with the resource denoted by this abstract path.
* <p>
* WMQFTE will read these properties to govern how a transfer behaves when
* interacting with the resource.
*
* @return A {@link IOExitProperties} instance for properties associated
*         with the resource denoted by this abstract path.
*/
IOExitProperties getProperties();
}

```

## Tâches associées

[Utilisation des exits utilisateur d'E-S de transfert MFT](#)

[Personnalisation de MFT avec des exits utilisateur](#)

## Interface *IOExitWildcardPath.java*

### IOExitWildcardPath.java

```
/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

/**
 * Represents a path that denotes a wildcard. This can be used to match multiple
 * resource paths.
 */
public interface IOExitWildcardPath extends IOExitPath {
```

## Tâches associées

[Utilisation des exits utilisateur d'E-S de transfert MFT](#)

[Personnalisation de MFT avec des exits utilisateur](#)

## Interface *MonitorExit.java*

### MonitorExit.java

```
/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * Copyright IBM Corp. 2009, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.util.Map;

/**
 * An interface that is implemented by classes that want to be invoked as part of
 * user exit routine processing. This interface defines a method that will be
 * invoked immediately prior to starting a task as the result of a monitor trigger
 */
public interface MonitorExit {

    /**
     * Invoked immediately prior to starting a task as the result of a monitor
     * trigger.
     *
     * @param environmentMetaData
     * meta data about the environment in which the implementation
     * of this method is running. This information can only be read,
     * it cannot be updated by the implementation. The constant
     * defined in EnvironmentMetaDataConstants class can
     * be used to access the data held by this map.
     */
}
```

```

*
* @param monitorMetaData
*      meta data to associate with the monitor. The meta data passed
*      to this method can be altered, and the changes will be
*      reflected in subsequent exit routine invocations. This map
*      also contains keys with IBM reserved names. These entries are
*      defined in the <code>MonitorMetaDataConstants</code> class and
*      have special semantics. The the values of the IBM reserved names
*      cannot be modified by the exit
*
* @param taskDetails
*      An XML String representing the task to be executed as a result of
*      the monitor triggering. This XML string may be modified by the
*      exit
*
* @return  a monitor exit result object which is used to determine if the
*          task should proceed, or be cancelled.
*/
MonitorExitResult onMonitor(Map<String, String> environmentMetaData,
                           Map<String, String> monitorMetaData,
                           Reference<String> taskDetails);
}

```

## Tâches associées

[Surveillance des ressources MFT](#)

[Personnalisation de MFT avec des exits utilisateur](#)

## Référence associée

[«Interface SourceTransferStartExit.java», à la page 2259](#)

[«Interface SourceTransferEndExit.java», à la page 2258](#)

[«Interface DestinationTransferStartExit.java», à la page 2235](#)

[«Interface DestinationTransferEndExit.java», à la page 2233](#)

[«Interface ProtocolBridgeCredentialExit.java», à la page 2254](#)

## *Interface ProtocolBridgeCredentialExit.java*

### ProtocolBridgeCredentialExit.java

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * © Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.util.Map;

/**
 * An interface that is implemented by classes that are to be invoked as part of
 * user exit routine processing. This interface defines methods that will
 * be invoked by a protocol bridge agent to map the MQ user ID of the transfer to credentials
 * that are to be used to access the protocol server.
 * There will be one instance of each implementation class per protocol bridge agent. The methods
 * can be called from different threads so the methods must be synchronized.
 */
public interface ProtocolBridgeCredentialExit {

    /**
     * Invoked once when a protocol bridge agent is started. It is intended to initialize
     * any resources that are required by the exit
     */
    * @param bridgeProperties
    *      The values of properties defined for the protocol bridge.
    *      These values can only be read, they cannot be updated by

```

```

*         the implementation.
*
*
* @return  true if the initialization is successful and false if unsuccessful
*         If false is returned from an exit the protocol bridge agent will not
*         start
*/

public boolean initialize(final Map<String>  bridgeProperties);

/**
 * Invoked once for each transfer to map the MQ user ID in the transfer message to the
 * credentials to be used to access the protocol server
 *
 * @param mqUserId The MQ user ID from which to map to the credentials to be used
 *               access the protocol server
 * @return         A credential exit result object that contains the result of the map and
 *               the credentials to use to access the protocol server
 */

public CredentialExitResult mapMQUserId(final String mqUserId);

/**
 * Invoked once when a protocol bridge agent is shutdown. It is intended to release
 * any resources that were allocated by the exit
 *
 * @param bridgeProperties
 *       The values of properties defined for the protocol bridge.
 *       These values can only be read, they cannot be updated by
 *       the implementation.
 *
 * @return
 */
*/

public void shutdown(final Map<String>  bridgeProperties);
}

```

## Tâches associées

[Personnalisation de MFT avec des exits utilisateur](#)

[Mappage des données d'identification pour un serveur de fichiers à l'aide de classes d'exit](#)

## ***Interface ProtocolBridgeCredentialExit2.java***

### **ProtocolBridgeCredentialExit2.java**

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * © Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

/**
 * An interface that is implemented by classes that are invoked as part of user
 * exit routine processing. This interface defines methods that are invoked by a
 * protocol bridge agent to map the MQ user ID of the transfer to credentials
 * used to access a specified protocol bridge server. There will be one instance
 * of each implementation class for each protocol bridge agent. The methods can
 * be called from different threads so the methods must be synchronized.
 */
public interface ProtocolBridgeCredentialExit2 extends
    ProtocolBridgeCredentialExit {

    /**
     * Invoked once for each transfer to map the MQ user ID in the transfer
     * message to the credentials used to access a specified protocol server.
     *
     */

```

```

    * @param endPoint
    *         Information that describes the protocol server to be accessed.
    * @param mqUserId
    *         The MQ user ID from which to map the credentials used to
    *         access the protocol server.
    * @return A {@link CredentialExitResult} instance that contains the result
    *         of the map and the credentials to use to access the protocol
    *         server.
    */
    public CredentialExitResult mapMQUserId(
        final ProtocolServerEndPoint endPoint, final String mqUserId);
}

```

### Tâches associées

Personnalisation de MFT avec des exits utilisateur

Mappage des données d'identification pour un serveur de fichiers à l'aide de classes d'exit

## Interface ProtocolBridgePropertiesExit2.java

### ProtocolBridgePropertiesExit2.java

```

/**
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * © Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.util.Map;
import java.util.Properties;

/**
 * An interface that is implemented by classes that are to be invoked as part of
 * user exit routine processing. This interface defines methods that will be
 * invoked by a protocol bridge agent to look up properties for protocol servers
 * that are referenced in transfers.
 * <p>
 * There will be one instance of each implementation class for each protocol
 * bridge agent. The methods can be called from different threads so the methods
 * must be synchronised.
 */
public interface ProtocolBridgePropertiesExit2 {

    /**
     * Invoked once when a protocol bridge agent is started. It is intended to
     * initialize any resources that are required by the exit.
     *
     * @param bridgeProperties
     *         The values of properties defined for the protocol bridge.
     *         These values can only be read, they cannot be updated by the
     *         implementation.
     * @return {@code true} if the initialization is successful and {@code
     *         false} if unsuccessful. If {@code false} is returned from an exit
     *         the protocol bridge agent will not start.
     */
    public boolean initialize(final Map<String, String> bridgeProperties);

    /**
     * Invoked when the Protocol Bridge needs to access the protocol bridge credentials XML file.
     *
     * @return a {@link String} object giving the location of the ProtocolBridgeCredentials.xml
     */
    public String getCredentialLocation ();

    /**
     * Obtains a set of properties for the specified protocol server name.
     * <p>
     * The returned {@link Properties} must contain entries with key names

```



```

* corresponding to the constants defined in
* {@link ProtocolServerPropertyConstants} and in particular must include an
* entry for all appropriate constants described as required.
*
* @param protocolServerName
*         The name of the protocol server whose properties are to be
*         returned. If a null or a blank value is specified, properties
*         for the default protocol server are to be returned.
* @return The {@link Properties} for the specified protocol server, or null
*         if the server cannot be found.
*/
public Properties getProtocolServerProperties(
    final String protocolServerName);

/**
* Invoked once when a protocol bridge agent is shut down. It is intended to
* release any resources that were allocated by the exit.
*
* @param bridgeProperties
*         The values of properties defined for the protocol bridge.
*         These values can only be read, they cannot be updated by the
*         implementation.
*/
public void shutdown(final Map<String, String> bridgeProperties);
}

```

### Tâches associées

[ProtocolBridgePropertiesExit: recherche des propriétés du serveur de fichiers de protocole](#)

[Personnalisation de MFT avec des exits utilisateur](#)

[Mappage des données d'identification pour un serveur de fichiers à l'aide de classes d'exit](#)

### **SourceFileExitFileSpecification.java**

#### SourceFileExitFileSpecification.java

```

/*
* Licensed Materials - Property of IBM
*
* "Restricted Materials of IBM"
*
* 5724-H72
*
* © Copyright IBM Corp. 2012, 2024. All Rights Reserved.
*
* US Government Users Restricted Rights - Use, duplication or
* disclosure restricted by GSA ADP Schedule Contract with
* IBM Corp.
*/
package com.ibm.wmqfte.exitroutine.api;

import java.util.Map;

/**
* A specification of the file names to use for a file transfer, as evaluated by the
* agent acting as the source of the transfer.
*/
public final class SourceFileExitFileSpecification {

    private final String sourceFileSpecification;
    private final String destinationFileSpecification;
    private final Map<String, String> sourceFileMetaData;
    private final Map<String, String> destinationFileMetaData;

    /**
    * Constructor. Creates a source file exit file specification.
    *
    * @param sourceFileSpecification
    *         the source file specification to associate with the source file
    *         exit file specification.
    *
    * @param destinationFileSpecification
    *         the destination file specification to associate with the
    *         source file exit file specification.
    *
    * @param sourceFileMetaData
    */
}

```

```

*           the source file meta data.
*
* @param destinationFileMetaData
*           the destination file meta data
*/
public SourceFileExitFileSpecification(final String sourceFileSpecification,
                                     final String destinationFileSpecification,
                                     final Map<String, String> sourceFileMetaData,
                                     final Map<String, String> destinationFileMetaData) {
    this.sourceFileSpecification = sourceFileSpecification;
    this.destinationFileSpecification = destinationFileSpecification;
    this.sourceFileMetaData = sourceFileMetaData;
    this.destinationFileMetaData = destinationFileMetaData;
}

/**
 * Returns the destination file specification.
 *
 * @return the destination file specification. This represents the location,
 *         on the agent acting as the destination for the transfer, where the
 *         file should be written. Exit routines installed into the agent
 *         acting as the destination for the transfer may override this value.
 */
public String getDestination() {
    return destinationFileSpecification;
}

/**
 * Returns the source file specification.
 *
 * @return the source file specification. This represents the location where
 *         the file data will be read from.
 */
public String getSource() {
    return sourceFileSpecification;
}

/**
 * Returns the file meta data that relates to the source file specification.
 *
 * @return the file meta data that relates to the source file specification.
 */
public Map<String, String> getSourceFileMetaData() {
    return sourceFileMetaData;
}

/**
 * Returns the file meta data that relates to the destination file specification.
 *
 * @return the file meta data that relates to the destination file specification.
 */
public Map<String, String> getDestinationFileMetaData() {
    return destinationFileMetaData;
}
}

```

## Concepts associés

«Métadonnées pour les exits utilisateur MFT», à la page 2221

Il existe trois types de métadonnées qui peuvent être fournis aux routines d'exit utilisateur pour Managed File Transfer: métadonnées d'environnement, de transfert et de fichier. Ces métadonnées sont présentées sous forme de mappes de paires clé-valeur Java .

## Interface SourceTransferEndExit.java

### SourceTransferEndExit.java

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * © Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or

```

```

* disclosure restricted by GSA ADP Schedule Contract with
* IBM Corp.
*/
package com.ibm.wmqfte.exitpoint.api;

/**
 * An interface that is implemented by classes that want to be invoked as part of
 * user exit routine processing. This interface defines a method that will be
 * invoked immediately after completing a transfer on the agent acting as the
 * source of the transfer.
 */
public interface SourceTransferEndExit {

    /**
     * Invoked immediately after the completion of a transfer on the agent acting as
     * the source of the transfer.
     *
     * @param transferExitResult
     *        a result object reflecting whether or not the transfer completed
     *        successfully.
     *
     * @param sourceAgentName
     *        the name of the agent acting as the source of the transfer.
     *        This is the name of the agent that the implementation of this
     *        method will be invoked from.
     *
     * @param destinationAgentName
     *        the name of the agent acting as the destination of the
     *        transfer.
     *
     * @param environmentMetaData
     *        meta data about the environment in which the implementation
     *        of this method is running. This information can only be read,
     *        it cannot be updated by the implementation. The constants
     *        defined in <code>EnvironmentMetaDataConstants</code> class can
     *        be used to access the data held by this map.
     *
     * @param transferMetaData
     *        meta data to associate with the transfer. The information can
     *        only be read, it cannot be updated by the implementation. This
     *        map may also contain keys with IBM reserved names. These
     *        entries are defined in the <code>TransferMetaDataConstants</code>
     *        class and have special semantics.
     *
     * @param fileResults
     *        a list of file transfer result objects that describe the source
     *        file name, destination file name and result of each file transfer
     *        operation attempted.
     *
     * @return
     *        an optional description to enter into the log message describing
     *        transfer completion. A value of <code>null</code> can be used
     *        when no description is required.
     */
    String onSourceTransferEnd(TransferExitResult transferExitResult,
                               String sourceAgentName,
                               String destinationAgentName,
                               Map<String, String>environmentMetaData,
                               Map<String, String>transferMetaData,
                               List<FileTransferResult>fileResults);

}

```

## Tâches associées

[Personnalisation de MFT avec des exits utilisateur](#)

## Référence associée

[«Interface SourceTransferStartExit.java», à la page 2259](#)

[«Interface DestinationTransferStartExit.java», à la page 2235](#)

[«Interface DestinationTransferEndExit.java», à la page 2233](#)

[«Interface MonitorExit.java», à la page 2253](#)

[«Interface ProtocolBridgeCredentialExit.java», à la page 2254](#)

## ***Interface SourceTransferStartExit.java***

## SourceTransferStartExit.java

```
/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * © Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitpoint.api;

import java.util.List;
import java.util.Map;

/**
 * An interface that is implemented by classes that want to be invoked as part of
 * user exit routine processing. This interface defines a method that will be
 * invoked immediately prior to starting a transfer on the agent acting as the
 * source of the transfer.
 */
public interface SourceTransferStartExit {

    /**
     * Invoked immediately prior to starting a transfer on the agent acting as
     * the source of the transfer.
     *
     * @param sourceAgentName
     *     the name of the agent acting as the source of the transfer.
     *     This is the name of the agent that the implementation of this
     *     method will be invoked from.
     *
     * @param destinationAgentName
     *     the name of the agent acting as the destination of the
     *     transfer.
     *
     * @param environmentMetaData
     *     meta data about the environment in which the implementation
     *     of this method is running. This information can only be read,
     *     it cannot be updated by the implementation. The constants
     *     defined in EnvironmentMetaDataConstants class can
     *     be used to access the data held by this map.
     *
     * @param transferMetaData
     *     meta data to associate with the transfer. The meta data passed
     *     to this method can be altered, and the changes to will be
     *     reflected in subsequent exit routine invocations. This map may
     *     also contain keys with IBM reserved names. These entries are
     *     defined in the TransferMetaDataConstants class and
     *     have special semantics.
     *
     * @param fileSpecs
     *     a list of file specifications that govern the file data to
     *     transfer. The implementation of this method can add entries,
     *     remove entries, or modify entries in this list and the changes
     *     will be reflected in the files transferred.
     *
     * @return
     *     a transfer exit result object which is used to determine if the
     *     transfer should proceed, or be cancelled.
     */
    TransferExitResult onSourceTransferStart(String sourceAgentName,
        String destinationAgentName,
        Map<String, String> environmentMetaData,
        Map<String, String> transferMetaData,
        List<SourceFileExitFileSpecification> fileSpecs);
}
```

### Tâches associées

[Personnalisation de MFT avec des exits utilisateur](#)

### Référence associée

«[SourceFileExitFileSpecification.java](#)», à la page 2257

[«Interface SourceTransferEndExit.java», à la page 2258](#)

[«Interface DestinationTransferStartExit.java», à la page 2235](#)

[«Interface DestinationTransferEndExit.java», à la page 2233](#)

[«Interface MonitorExit.java», à la page 2253](#)

[«Interface ProtocolBridgeCredentialExit.java», à la page 2254](#)

## ***Interface TransferExitResult.java***

### **TransferExitResult.java**

```
/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * © Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */

package com.ibm.wmqfte.exitroutine.api;

/**
 * The result of invoking a transfer exit routine. It is composed of a result
 * code, which determines if the transfer should proceed, and an optional explanatory
 * message. The explanation, if present, is entered into the log message.
 */
public class TransferExitResult {

    private final TransferExitResultCode resultCode;
    private final String explanation;

    /**
     * For convenience, a static "proceed" result with no associated explanation
     * message.
     */
    public static final TransferExitResult PROCEED_RESULT =
        new TransferExitResult(TransferExitResultCode.PROCEED, null);

    /**
     * Constructor. Creates a transfer exit result object with a specified result
     * code and explanation.
     *
     * @param resultCode
     *         The result code to associate with the exit result being created.
     *
     * @param explanation
     *         The explanation to associate with the exit result being created.
     *         A value of <code>null</code> can be specified to indicate no
     *         explanation.
     */
    public TransferExitResult(TransferExitResultCode resultCode, String explanation) {
        this.resultCode = resultCode;
        this.explanation = explanation;
    }

    /**
     * Returns the explanation associated with this transfer exit result.
     *
     * @return
     *         the explanation associated with this exit result.
     */
    public String getExplanation() {
        return explanation;
    }

    /**
     * Returns the result code associated with this transfer exit result.
     *
     * @return
     *         the result code associated with this exit result.
     */
}
```

```
public TransferExitResultCode getResultCode() {
    return resultCode;
}
}
```

### Tâches associées

Personnalisation de MFT avec des exits utilisateur

### Référence associée

«[Interface SourceTransferStartExit.java](#)», à la page 2259

«[Interface DestinationTransferStartExit.java](#)», à la page 2235

«[Interface DestinationTransferEndExit.java](#)», à la page 2233

«[Interface MonitorExit.java](#)», à la page 2253

«[Interface ProtocolBridgeCredentialExit.java](#)», à la page 2254

## Formats de message pour les messages que vous pouvez placer dans la file d'attente de commandes de l'agent MFT

Ces schémas XML définissent les formats des messages qui peuvent être placés dans la file d'attente de commandes de l'agent pour demander à l'agent d'effectuer une action. Le message XML peut être placé dans la file d'attente de commandes de l'agent à l'aide des commandes de ligne de commande ou par une application.

- [Format de message de demande de transfert de fichier](#)
- [Formats de message de demande du moniteur MFT](#)
- [Format de message de demande de l'agent Ping MFT](#)
- [Format de message de réponse de l'agent MFT](#)

## Informations de référence sur la messagerie REST API

Cette rubrique fournit des informations de référence sur le messaging REST API.

Pour plus d'informations sur l'utilisation de messaging REST API, voir [Messagerie à l'aide de REST API](#).

### ressources REST API

Cette collection de rubriques fournit des informations de référence pour chacune des ressources messaging REST API .

Pour plus d'informations sur l'utilisation de messaging REST API, voir [Messagerie à l'aide de REST API](#).

#### ***/messaging/qmgr/{qmgrName}/queue/{queueName}/message***

L'API REST de messagerie permet d'insérer des messages dans une file d'attente ou de les parcourir ou de les obtenir de façon destructive à partir d'une file d'attente, à l'aide de la ressource `/messaging/qmgr/{qmgrName}/queue/{queueName}/message` .

#### ***POST /messaging/qmgr/{qmgrName}/queue/{queueName}/message***

Vous pouvez utiliser la méthode POST HTTP avec la ressource `/messaging/qmgr/{qmgrName}/queue/{queueName}/message` pour insérer des messages dans la file d'attente spécifiée sur le gestionnaire de files d'attente spécifié.

Insère un message IBM MQ contenant le corps de demande HTTP dans le gestionnaire de files d'attente et la file d'attente spécifiés. Le gestionnaire de files d'attente doit se trouver sur la même machine que le serveur mqweb. La méthode prend uniquement en charge les corps de demande HTTP basés sur du

texte. Les messages sont envoyés sous forme de messages au format MQSTR ou JMS `TextMessage` et sont insérés à l'aide du contexte utilisateur en cours.

**V 9.3.0** L'API REST V3 permet de spécifier des propriétés de message définies par l'utilisateur et d'inclure la priorité des messages. Les en-têtes de demande `ibm-mq-md-priority` et `ibm-mq-usr` ne sont disponibles qu'avec l'API REST V3. L'en-tête de demande `ibm-mq-md-correlationId` a un format différent dans l'API REST V3. L'en-tête peut être un ID spécifique à l'application ou, s'il s'agit d'une chaîne codée, requiert le préfixe `ID:` . Si votre demande POST contient des messages définis par l'utilisateur ou un ID de corrélation spécifique à l'application, le message est formaté en tant que JMS `TextMessage`.

- «URL de ressource», à la page 2263
- «En-têtes de demande :», à la page 2264
- «Format du corps de la demande», à la page 2266
- «Sécurité», à la page 2266
- «Codes de statut de réponse», à la page 2267
- «En-têtes de réponse :», à la page 2267
- «Format du corps de la réponse», à la page 2268
- «Exemples», à la page 2268

## URL de ressource

`https://host:port/ibmmq/rest/v1/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

`https://host:port/ibmmq/rest/v2/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

**V 9.3.0** `https://host:port/ibmmq/rest/v3/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

### qmgrName

Indique le nom du gestionnaire de files d'attente auquel se connecter pour la messagerie. Le gestionnaire de files d'attente doit se trouver sur la même machine que le serveur mqweb.

**V 9.3.3** Depuis IBM MQ 9.3.3, vous pouvez vous connecter à un gestionnaire de files d'attente local ou à un gestionnaire de files d'attente éloignées. Le nom que vous spécifiez pour **qmgrName** dépend de la manière dont votre serveur mqweb est configuré :

- Si votre serveur mqweb est configuré pour se connecter uniquement aux gestionnaires de files d'attente locaux, utilisez le nom du gestionnaire de files d'attente.
- Si votre serveur mqweb est configuré pour connecter des gestionnaires de files d'attente éloignées, utilisez le nom unique du gestionnaire de files d'attente comme indiqué dans les informations de connexion du gestionnaire de files d'attente éloignées.

Vous pouvez déterminer si votre serveur mqweb est configuré pour se connecter à des gestionnaires de files d'attente locaux ou à des gestionnaires de files d'attente éloignées à l'aide de la commande **dspmweb properties** et en affichant la propriété **mqRestMessagingConnectionMode** .

Le nom du gestionnaire de files d'attente est sensible à la casse.

Si le nom du gestionnaire de files d'attente inclut une barre oblique, un point ou le signe de pourcentage, ces caractères doivent être codés dans l'URL :

- Une barre oblique doit être codée en tant que %2F.
- Un point doit être codé en tant que %2E.
- Un signe de pourcentage doit être codé en tant que %25.

### queueName

Indique le nom de la file d'attente dans laquelle le message doit être inséré.

La file d'attente doit être définie comme étant locale, éloignée ou alias du gestionnaire de files d'attente spécifié ; elle peut également faire référence à une file d'attente en cluster.

Le nom de la file d'attente est sensible à la casse.

Si le nom de la file d'attente contient une barre oblique ou un signe de pourcentage, les caractères suivants doivent être codés URL :

- Une barre oblique (/) doit être codée sous la forme %2F.
- Le signe de pourcentage,% , doit être codé en tant que %25.

Vous pouvez utiliser HTTP à la place de HTTPS si vous activez les connexions HTTP . Pour plus d'informations sur l'activation de HTTP, voir [Configuration des ports HTTP et HTTPS](#).

## En-têtes de demande :

Les en-têtes suivants doivent être envoyés avec la demande:

### Autorisation

Cet en-tête doit être envoyé si vous utilisez l'authentification de base. Pour plus d'informations, voir [Using HTTP basic authentication with the REST API](#).

### Content-Type

Cet en-tête doit être envoyé avec l'une des valeurs suivantes:

- text/plain;charset=utf-8
- text/html;charset=utf-8
- text/xml;charset=utf-8
- application/json;charset=utf-8
- application/xml;charset=utf-8

### ibm-mq-rest-csrf-token

Cet en-tête doit être défini, mais sa valeur est indifférente, et peut notamment être vide.

Les en-têtes suivants peuvent éventuellement être envoyés avec la demande:

### Langage d'acceptation

Cet en-tête indique la langue requise pour les exceptions ou les messages d'erreur renvoyés dans le corps du message de réponse.

#### **ibm-mq-md-correlationId**

Cet en-tête définit l'ID de corrélation du message créé. L'en-tête est indiqué sous la forme d'une chaîne codée hexadécimale de 48 caractères représentant 24 octets. Ne préfixez pas la valeur avec "ID: ", l'API REST ajoute cette chaîne automatiquement.

Par exemple: `ibm-mq-md-correlationId: 414d5120514d41444556202020202067d8bf5923582e02`

#### **ibm-mq-md-correlationId**

Cet en-tête définit l'ID de corrélation du message créé. L'ID corrélation peut prendre l'une des formes suivantes:

- Chaîne codée en hexadécimal de 48 caractères, représentant 24 octets, précédée de la chaîne "ID: ". Par exemple : `ibm-mq-md-correlationId: ID:414d5120514d41444556202020202067d8bf5923582e02`
- Valeur spécifique à l'application. La valeur est une chaîne spécifique à l'application: `ibm-mq-md-correlationId: My-Custom-CorrelId`

Si vous spécifiez cette forme d'ID de corrélation, la destination de message est ciblée en tant que `WMQ_CLIENT_JMS_???` et intègre donc un en-tête `MQRFH2` .



### **ibm-mq-md-expiration**

Cet en-tête définit la durée d'expiration du message créé. L'expiration d'un message commence à partir du moment où le message arrive dans la file d'attente. Par conséquent, le temps d'attente du réseau est ignoré. L'en-tête doit être spécifié sous la forme de l'une des valeurs suivantes:

#### **illimité**

Le message n'expire pas.

Cette valeur est la valeur par défaut.

#### **Valeur entière**

Millisecondes avant l'expiration du message.

Limité à la plage de 0 à 99999999900.

### **ibm-mq-md-persistence**

Cet en-tête définit la persistance du message créé. L'en-tête est spécifié sous la forme de l'une des valeurs suivantes:

#### **nonPersistent**

Le message ne survit pas aux défaillances du système ou aux redémarrages du gestionnaire de files d'attente.

Cette valeur est la valeur par défaut.

#### **persistant**

Le message survit aux échecs du système ou aux redémarrages du gestionnaire de files d'attente.

V 9.3.0

REST API V3

### **ibm-mq-md-priority**

Pour l'API REST V3, cet en-tête définit la priorité du message créé. L'en-tête est spécifié sous la forme de l'une des valeurs suivantes:

#### **asDestination**

Le message utilise la priorité spécifiée dans l'attribut DEFPTY de l'objet de file d'attente IBM MQ sous-jacent.

Cette valeur est la valeur par défaut.

#### **Valeur entière**

Indiquez la priorité réelle sous la forme d'un entier compris entre 0 et 9.

Par exemple: `ibm-mq-md-priority: asDestination`

Pour l'API REST V1 et l'API REST V2, la priorité des messages pour POST est toujours 4.

### **ibm-mq-md-replyTo**

Cet en-tête définit la destination de réponse pour le message créé. Le format de l'en-tête utilise la notation standard consistant à fournir la file d'attente de réponse et un gestionnaire de files d'attente facultatif: `replyQueue[@replyQmgr]`

Par exemple: `ibm-mq-md-replyTo: myReplyQueue@myReplyQMGR`

V 9.3.0

REST API V3

### **ibm-mq-usr**

Cet en-tête définit les propriétés définies par l'utilisateur du message de demande.

La syntaxe des propriétés est la suivante:

`ibm-mq-usr: property_name; user_value; user_type`

#### **nom\_propriété**

Nom de la propriété utilisateur spécifiée. Il doit s'agir d'un nom de propriété JMS valide.

#### **valeur\_utilisateur**

Valeur de la propriété.

#### **type\_utilisateur**

Type de la propriété:

- boolean (true/false, MQBOOL)
- byte (entier 8 bits, MQINT8)

- `short` (entier 16 bits, MQINT16)
- `integer` (entier 32 bits, MQINT32)
- `long` (entier 64 bits, MQINT64)
- `float` (réel 32 bits, MQFLOAT32)
- `double` (réel 64 bits, MQFLOAT64)
- `string` (chaîne entre guillemets)


Vous pouvez définir plusieurs propriétés sur un message. Vous pouvez spécifier plusieurs propriétés séparées par des virgules dans un seul en-tête de demande `ibm-mq-usr` ou utiliser deux ou plusieurs instances distinctes de l'en-tête de demande `ibm-mq-usr`.

Par exemple, vous pouvez définir plusieurs propriétés définies par l'utilisateur dans un même en-tête:  
`ibm-mq-usr: userPropertyA;5;byte,userPropertyB;-10;integer`

Vous pouvez également définir plusieurs propriétés définies par l'utilisateur dans des instances distinctes de l'en-tête:  
`ibm-mq-usr: userPropertyA;5;byte` `ibm-mq-usr: userPropertyB;-10;integer`

## Format du corps de la demande

Le corps de la demande doit être du texte et utiliser le codage UTF-8 . Aucune structure de texte spécifique n'est requise. Un message au format MQSTR contenant le texte du corps de la demande est créé et inséré dans la file d'attente spécifiée.

 Si les propriétés définies par l'utilisateur de l'API REST V3 ou les fonctions d'ID de corrélation spécifiques à l'application sont utilisées, un message au format JMS `TextMessage` contenant le texte du corps de la demande est créé et placé dans la file d'attente spécifiée.



Pour plus d'informations, voir [Exemples](#).

## Sécurité


L'appelant doit être authentifié auprès du serveur mqweb. Les rôles MQWebAdmin et MQWebAdminRO ne sont pas applicables pour le messaging REST API. Pour plus d'informations sur la sécurité pour REST API, voir [Sécurité IBM MQ Console et REST API](#).

Une fois authentifié auprès du serveur mqweb, l'utilisateur est capable d'utiliser à la fois messaging REST API et administrative REST API.

Le principal de sécurité de l'appelant doit être autorisé à insérer des messages dans la file d'attente spécifiée:

- La file d'attente spécifiée par la partie `{queueName}` de l'URL de ressource doit être activée pour PUT.
-  Pour la file d'attente spécifiée par la partie `{queueName}` de la ressource URL, les droits +PUT doivent être accordés au principal de sécurité de l'appelant.
-  Pour la file d'attente spécifiée par la partie `{queueName}` de la ressource URL, l'accès UPDATE doit être accordé au principal de sécurité de l'appelant.

Ce principal de sécurité peut être l'utilisateur qui a démarré le serveur mqweb ou l'utilisateur qui est connecté au serveur mqweb. Si le gestionnaire de files d'attente auquel vous vous connectez est un gestionnaire de files d'attente éloignées, le principal de sécurité peut être l'utilisateur fourni par les données d'identification dans les informations de connexion du gestionnaire de files d'attente éloignées ou un autre utilisateur déterminé par les règles de sécurité du canal. Pour plus d'informations, voir [Détermination du principal de sécurité utilisé par messaging REST API](#).

 Sous AIX, Linux, and Windows, vous pouvez accorder des droits aux principaux de sécurité pour utiliser les ressources IBM MQ à l'aide de la commande **setmqaut**. Pour plus d'informations, voir [setmqaut \(grant of revoke authority\)](#).

Si vous utilisez Advanced Message Security (AMS) avec messaging REST API, notez que tous les messages sont chiffrés à l'aide du contexte du serveur mqweb et non du contexte de l'utilisateur qui a envoyé le message.

## Codes de statut de réponse

### 201

Le message a été créé et envoyé.

### 400

Des données non valides ont été fournies.

Par exemple, une valeur d'en-tête de demande non valide a été spécifiée.

### 401

Non authentifié.

L'appelant doit être authentifié sur le serveur mqweb et doit être membre d'un ou plusieurs rôles MQWebAdmin, MQWebAdminRO ou MQWebUser. L'en-tête `ibm-mq-rest-csrf-token` doit également être spécifié. Pour plus d'informations, voir «Sécurité», à la page 2266.

### 403

Non autorisé

L'appelant est authentifié auprès du serveur mqweb et est associé à un principal valide. Toutefois, le principal n'a pas accès à l'ensemble ou à un sous-ensemble des ressources IBM MQ requises ou n'est pas dans le rôle MQWebUser. Pour plus d'informations sur l'accès requis, voir «Sécurité», à la page 2266.

### 404

La file d'attente n'existe pas.

### 405

La file d'attente est interdite par la commande PUT.

### 415

Un en-tête ou un corps de message est un type de support non pris en charge.

Par exemple, l'en-tête `Content-Type` est défini sur un type de support non pris en charge.

### 500

Problème de serveur ou code d'erreur provenant de IBM MQ.

### 502

Le principal de sécurité en cours ne peut pas envoyer le message car le fournisseur de messagerie ne prend pas en charge la fonction requise. Par exemple, si le chemin d'accès aux classes du serveur mqweb n'est pas valide.

### 503

Gestionnaire de files d'attente non en cours d'exécution.

## En-têtes de réponse :

Les en-têtes suivants sont renvoyés avec la réponse:

### Contenu-Langue

Indique l'identificateur de langue du message de réponse en cas d'erreurs ou d'exceptions. Utilisé conjointement avec l'en-tête de demande `Accept-Language` pour indiquer la langue requise pour toute condition d'erreur ou d'exception. La valeur par défaut du serveur mqweb est utilisée si la langue demandée n'est pas prise en charge.

### Longueur de contenu

Indique la longueur du corps de la réponse HTTP, même en l'absence de contenu. En cas de réussite, la valeur est égale à zéro.

## Content-Type

Indique le type de corps de réponse. En cas de réussite, la valeur est `text/plain; charset=utf-8`. En cas d'erreurs ou d'exceptions, la valeur est `application/json; charset=utf-8`.

### REST API V1 > REST API V2 **ibm-mq-md-messageId**

Indique l'ID de message attribué par IBM MQ à ce message. Tout comme l'en-tête de demande `ibm-mq-md-correlationId`, il est représenté sous la forme d'une chaîne hexadécimale de 48 caractères représentant 24 octets.

Exemple :

```
ibm-mq-md-messageId: 414d5120514d4144455620202020202067d8ce5923582f07
```

### V 9.3.0 > REST API V3 **ibm-mq-md-messageId**

Indique l'ID de message attribué par IBM MQ à ce message. Tout comme l'en-tête de demande `ibm-mq-md-correlationId`, il est représenté sous la forme d'une chaîne codée hexadécimale de 48 caractères, représentant 24 octets, préfixée par la chaîne `ID:`.

Exemple :

```
ibm-mq-md-messageId: ID:414d5120514d4144455620202020202067d8ce5923582f07
```

### V 9.3.3 **ibm-mq-resolved-qmgr**

Indique le nom du gestionnaire de files d'attente utilisé pour la demande. Si un nom unique a été utilisé dans la ressource URL, cet en-tête identifie le nom du gestionnaire de files d'attente associé à ce nom unique. Si le nom unique utilisé dans la ressource URL fait référence à un groupe de gestionnaires de files d'attente, cet en-tête identifie le gestionnaire de files d'attente utilisé dans le groupe.

## Format du corps de la réponse

Le corps de la réponse est vide si le message est envoyé avec succès. Si une erreur se produit, le corps de la réponse contient un message d'erreur. Pour plus d'informations, voir [Traitement des erreurs REST API](#).

## Exemples

Les exemples ci-après utilisent l'URL de ressource de version 2. Si vous utilisez une version de IBM MQ antérieure à IBM MQ 9.1.5, vous devez utiliser l'URL de ressource v1 à la place. En d'autres cas, dans l'URL de ressource, remplacez v1 où l'exemple d'URL utilise v2.

L'exemple suivant permet de se connecter à un utilisateur appelé `muser` avec le mot de passe `muser`. Dans cURL, la demande de connexion peut ressembler à l'exemple Windows suivant. Le jeton LTPA est stocké dans le fichier `cookiejar.txt` à l'aide de l'indicateur `-c` :

```
curl -k "https://localhost:9443/ibmmq/rest/v2/login" -X POST
-H "Content-Type: application/json" --data "{\"username\": \"muser\", \"password\": \"muser\"}"
-c c:\cookiejar.txt
```

Une fois l'utilisateur connecté, le jeton LTPA et l'en-tête `ibm-mq-rest-csrf-token` HTTP sont utilisés pour authentifier d'autres demandes. Le `ibm-mq-rest-csrf-token` `token_value` peut être n'importe quelle valeur, y compris vide.

- L'exemple Windows cURL suivant envoie un message à la file d'attente Q1 sur le gestionnaire de files d'attente QM1 à l'aide des options par défaut. Le message contient le texte `"Hello World !"`:

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/Q1/message"
-X POST -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token_value"
-H "Content-Type: text/plain; charset=utf-8" --data "Hello World!"
```

- L'exemple Windows cURL suivant envoie un message persistant à la file d'attente Q1 sur le gestionnaire de files d'attente QM1, avec un délai d'expiration de 2 minutes. Le message contient le texte "Hello World !":

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/Q1/message"
-X POST -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token_value"
-H "Content-Type: text/plain;charset=utf-8" -H "ibm-mq-md-persistence: persistent"
-H "ibm-mq-md-expiry: 120000" --data "Hello World!"
```

- L'exemple Windows cURL suivant envoie un message non persistant à la file d'attente Q1 du gestionnaire de files d'attente QM1, sans expiration et avec un ID de corrélation défini. Le message contient le texte "Hello World !":

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/Q1/message"
-X POST -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token-value"
-H "Content-Type: text/plain;charset=utf-8" -H "ibm-mq-md-persistence: nonPersistent"
-H "ibm-mq-md-expiry: unlimited" -H "ibm-mq-md-correlationId:
414d5120514d4144455620202020202067d8b
f5923582e02" --data "Hello World!"
```

### **GET/messaging/qmgr/{qmgrName}/queue/{queueName}/message**

Vous pouvez utiliser la méthode HTTP GET avec la ressource /messaging/qmgr/{qmgrName}/queue/{queueName}/message pour parcourir les messages du gestionnaire de files d'attente et de la file d'attente associés.

Parcourt le premier message disponible à partir du gestionnaire de files d'attente et de la file d'attente spécifiés. Le gestionnaire de files d'attente doit se trouver sur la même machine que le serveur mqweb. Le corps du message est renvoyé dans le corps de réponse HTTP . Le message doit être au format MQSTR ou JMS TextMessage et il est reçu à l'aide du contexte utilisateur en cours.

Tous les messages sont laissés dans la file d'attente et un code d'état approprié est renvoyé à l'appelant pour les messages inappropriés. Par exemple, un message qui n'a pas le format MQSTR ou JMS TextMessage .

**V 9.3.0** L'API REST V3 ajoute la possibilité de spécifier des propriétés de message définies par l'utilisateur et d'inclure la priorité des messages dans les messages. Les en-têtes de réponse ibm-mq-md-priority et ibm-mq-usr ne sont disponibles qu'avec l'API REST V3. L'en-tête de demande ibm-mq-md-correlationId a un format différent dans l'API REST V3. L'en-tête peut être un ID spécifique à l'application ou, s'il s'agit d'une chaîne codée, il conserve le préfixe ID : . L'en-tête de réponse et le paramètre de requête ibm-mq-md-messageId ont un format différent dans l'API REST V3 et conservent le préfixe ID : .

- [«URL de ressource»](#), à la page 2269
- [«Paramètres de requête facultatifs»](#), à la page 2270
- [«En-têtes de demande :»](#), à la page 2271
- [«Format du corps de la demande»](#), à la page 2271
- [«Sécurité»](#), à la page 2272
- [«Codes de statut de réponse»](#), à la page 2272
- [«En-têtes de réponse :»](#), à la page 2273
- [«Format du corps de la réponse»](#), à la page 2275
- [«Exemples»](#), à la page 2275

### **URL de ressource**

`https://host:port/ibmmq/rest/v1/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

`https://host:port/ibmmq/rest/v2/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

**V 9.3.0** `https://host:port/ibmmq/rest/v3/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

### qmgrName

Indique le nom du gestionnaire de files d'attente auquel se connecter pour la messagerie. Le gestionnaire de files d'attente doit se trouver sur la même machine que le serveur mqweb.

**V 9.3.3** Depuis IBM MQ 9.3.3, vous pouvez vous connecter à un gestionnaire de files d'attente local ou à un gestionnaire de files d'attente éloignées. Le nom que vous spécifiez pour **qmgrName** dépend de la manière dont votre serveur mqweb est configuré:

- Si votre serveur mqweb est configuré pour se connecter uniquement aux gestionnaires de files d'attente locaux, utilisez le nom du gestionnaire de files d'attente.
- Si votre serveur mqweb est configuré pour connecter des gestionnaires de files d'attente éloignées, utilisez le nom unique du gestionnaire de files d'attente comme indiqué dans les informations de connexion du gestionnaire de files d'attente éloignées.

Vous pouvez déterminer si votre serveur mqweb est configuré pour se connecter à des gestionnaires de files d'attente locaux ou à des gestionnaires de files d'attente éloignées à l'aide de la commande **dspmqweb properties** et en affichant la propriété **mqRestMessagingConnectionMode**.

Le nom du gestionnaire de files d'attente est sensible à la casse.

Si le nom du gestionnaire de files d'attente inclut une barre oblique, un point ou le signe de pourcentage, ces caractères doivent être codés dans l'URL :

- Une barre oblique (/) doit être codée sous la forme %2F.
- Le signe pourcentage (%) doit être codé en tant que %25.

### queueName

Indique le nom de la file d'attente à partir de laquelle parcourir le message.

La file d'attente doit être définie comme étant locale ou un alias pointant vers une file d'attente locale.

Le nom de la file d'attente est sensible à la casse.

Si le nom de la file d'attente contient une barre oblique ou un signe de pourcentage, les caractères suivants doivent être codés URL :

- Une barre oblique (/) doit être codée sous la forme %2F.
- Le signe de pourcentage, %, doit être codé en tant que %25.

Vous pouvez utiliser HTTP à la place de HTTPS si vous activez les connexions HTTP . Pour plus d'informations sur l'activation de HTTP, voir [Configuration des ports HTTP et HTTPS](#).

## Paramètres de requête facultatifs

**REST API V1** **REST API V2** **correlationId=hexValue**

Indique que la méthode HTTP renvoie le message suivant avec l'ID de corrélation correspondant.

### hexValue

Le paramètre de requête doit être indiqué sous la forme d'une chaîne codée hexadécimale de 48 caractères représentant 24 octets.

Exemple :

```
../message?correlationId=414d5120514d4144455620202020202067d8bf5923582e02
```

**V 9.3.0** **REST API V3** **correlationId= ID:hexValue ou correlationId=valeur\_spécifique\_application**

Indique que la méthode HTTP renvoie le message suivant avec l'ID de corrélation correspondant.

### hexValue

Le paramètre de requête doit être indiqué sous la forme d'une chaîne codée hexadécimale de 48 caractères, précédée de la chaîne "ID: ".

Exemple :

```
../message?correlationId=ID:414d5120514d41444556202020202067d8bf5923582e02
```

### valeur\_spécif\_application

Le paramètre de requête peut être spécifié en tant que chaîne spécifique à l'application.

Exemple :

```
../message?correlationId=My-Custom-CorrelId
```

### REST API V1 > REST API V2 messageId=hexValue

Indique que la méthode HTTP renvoie le message suivant avec l'ID de message correspondant.

#### hexValue

Le paramètre de requête doit être indiqué sous la forme d'une chaîne codée hexadécimale de 48 caractères représentant 24 octets.

Exemple :

```
../message?messageId=414d5120514d41444556202020202067d8ce5923582f07
```

### V 9.3.0 > REST API V3 messageId= ID:hexValue

Indique que la méthode HTTP renvoie le message suivant avec l'ID de message correspondant.

#### hexValue

Le paramètre de requête doit être indiqué sous la forme d'une chaîne codée hexadécimale de 48 caractères, représentant 24 octets, précédée de la chaîne "ID :".

Exemple :

```
../message?messageId=ID:414d5120514d41444556202020202067d8ce5923582f07
```

## En-têtes de demande :

Les en-têtes suivants doivent être envoyés avec la demande:

### Autorisation

Cet en-tête doit être envoyé si vous utilisez l'authentification de base. Pour plus d'informations, voir [Using HTTP basic authentication with the REST API](#).

### ibm-mq-rest-csrf-token

Cet en-tête doit être défini, mais sa valeur est indifférente, et peut notamment être vide.

Les en-têtes suivants peuvent éventuellement être envoyés avec la demande:

### Accepter-Ensemble de caractères

Cet en-tête peut être utilisé pour indiquer quel jeu de caractères est acceptable pour la réponse. S'il est spécifié, cet en-tête doit être défini sur UTF-8.

### Langage d'acceptation

Cet en-tête indique la langue requise pour les exceptions ou les messages d'erreur renvoyés dans le corps du message de réponse.

## Format du corps de la demande




Néant.

## Sécurité


L'appelant doit être authentifié auprès du serveur mqweb. Les rôles MQWebAdmin et MQWebAdminRO ne sont pas applicables pour le messaging REST API. Pour plus d'informations sur la sécurité pour REST API, voir [Sécurité IBM MQ Console et REST API](#).

Une fois authentifié auprès du serveur mqweb, l'utilisateur est capable d'utiliser à la fois messaging REST API et administrative REST API.

Le principal de sécurité de l'appelant doit avoir le droit de parcourir les messages dans la file d'attente spécifiée :

- La file d'attente spécifiée par la partie *{queueName}* de l'URL de ressource doit être activée pour BROWSE.
-   Pour la file d'attente spécifiée par la partie *{queueName}* de l'URL de ressource, les droits +GET, +INQ et +BROWSE doivent être accordés au principal de sécurité de l'appelant.
-  Pour la file d'attente spécifiée par la partie *{queueName}* de l'URL de ressource, UPDATE, l'accès doit être accordé au principal de sécurité de l'appelant.

Ce principal de sécurité peut être l'utilisateur qui a démarré le serveur mqweb ou l'utilisateur qui est connecté au serveur mqweb. Si le gestionnaire de files d'attente auquel vous vous connectez est un gestionnaire de files d'attente éloignées, le principal de sécurité peut être l'utilisateur fourni par les données d'identification dans les informations de connexion du gestionnaire de files d'attente éloignées ou un autre utilisateur déterminé par les règles de sécurité du canal. Pour plus d'informations, voir [Détermination du principal de sécurité utilisé par messaging REST API](#).

 Sous AIX, Linux, and Windows, vous pouvez accorder des droits aux principaux de sécurité pour utiliser les ressources IBM MQ à l'aide de la commande **setmqaut**. Pour plus d'informations, voir [setmqaut](#) (grant of revoke authority).

 Sous z/OS, voir [Configuration de la sécurité sous z/OS](#).

## Codes de statut de réponse

### 200

Le message a été reçu.

### 204

Aucun message disponible.

### 400

Des données non valides ont été fournies.

Par exemple, une valeur de paramètre de requête non valide a été spécifiée.

### 401

Non authentifié.

L'appelant doit être authentifié sur le serveur mqweb et doit être membre d'un ou plusieurs rôles MQWebAdmin, MQWebAdminRO ou MQWebUser. L'en-tête `ibm-mq-rest-csrf-token` doit également être spécifié. Pour plus d'informations, voir [«Sécurité»](#), à la page 2272.

### 403

Non autorisé

L'appelant est authentifié auprès du serveur mqweb et est associé à un principal valide. Toutefois, le principal n'a pas accès à l'ensemble ou à un sous-ensemble des ressources IBM MQ requises ou n'est pas dans le rôle MQWebUser. Pour plus d'informations sur l'accès requis, voir [«Sécurité»](#), à la page 2272.

### 404

La file d'attente n'existe pas.



**500**

Problème de serveur ou code d'erreur provenant de IBM MQ.

**501**

La réponse HTTP n'a pas pu être construite.

Par exemple, le type du message reçu est incorrect ou correct, mais le corps n'a pas pu être traité.

**502**

Le principal de sécurité en cours ne peut pas recevoir le message car le fournisseur de messagerie ne prend pas en charge la fonction requise. Par exemple, si le chemin d'accès aux classes du serveur mqweb n'est pas valide.

**503**

Gestionnaire de files d'attente non en cours d'exécution.

**En-têtes de réponse :**

Les en-têtes suivants sont renvoyés avec la réponse:

**Contenu-Langue**

Indique l'identificateur de langue du message de réponse en cas d'erreurs ou d'exceptions. Utilisé conjointement avec l'en-tête de demande Accept-Language pour indiquer la langue requise pour toute condition d'erreur ou d'exception. La valeur par défaut du serveur mqweb est utilisée si la langue demandée n'est pas prise en charge.

**Longueur de contenu**

Indique la longueur du corps de la réponse HTTP, même en l'absence de contenu. La valeur contient la longueur (octets) des données de message.

**Content-Type**

Indique le type de contenu renvoyé dans le corps de réponse du message reçu. En cas de réussite, la valeur est `text/plain; charset=utf-8`. En cas d'erreurs ou d'exceptions, la valeur est `application/json; charset=utf-8`.

**REST API V1 > REST API V2 `ibm-mq-md-correlationId`**

Indique l'ID de corrélation du message reçu. L'en-tête est renvoyé si le message reçu contient un ID de corrélation valide. Elle est représentée sous la forme d'une chaîne codée hexadécimale de 48 caractères, représentant 24 octets.

Exemple :

```
ibm-mq-md-correlationId: 414d5120514d414445562020202020202067d8bf5923582e02
```

**V 9.3.0 > REST API V3 `ibm-mq-md-correlationId`**

Indique l'ID de corrélation du message reçu. L'en-tête est renvoyé si le message reçu contient un ID de corrélation valide. L'ID corrélation peut prendre l'une des formes suivantes:

- Chaîne codée en hexadécimal de 48 caractères, représentant 24 octets, précédée de la chaîne "ID:". Exemple :

```
ibm-mq-md-correlationId: ID:414d5120514d414445562020202020202067d8bf5923582e02
```

- Valeur spécifique à l'application. La valeur est une chaîne spécifique à l'application:

```
ibm-mq-md-correlationId: My-Custom-CorrelId
```

**ibm-mq-md-expiration**

Indique la durée d'expiration restante du message reçu. L'en-tête peut avoir l'une des valeurs suivantes:

**illimité**

Le message n'expire pas.

### **Valeur entière**

Nombre de millisecondes restant avant l'expiration du message.

#### **REST API V1** **REST API V2** **ibm-mq-md-messageId**

Indique l'ID de message attribué par IBM MQ à ce message. Tout comme l'en-tête `ibm-mq-md-correlationId`, il est représenté sous la forme d'une chaîne codée hexadécimale de 48 caractères représentant 24 octets.

Exemple :

```
ibm-mq-md-messageId: 414d5120514d4144455620202020202067d8ce5923582f07
```

#### **V 9.3.0** **REST API V3** **ibm-mq-md-messageId**

Indique l'ID de message attribué par IBM MQ à ce message. Comme l'en-tête `ibm-mq-md-correlationId`, il est représenté sous la forme d'une chaîne codée hexadécimale de 48 caractères, représentant 24 octets, préfixée avec la chaîne "ID: "

Exemple :

```
ibm-mq-md-messageId: ID:414d5120514d4144455620202020202067d8ce5923582f07
```

### **ibm-mq-md-persistence**

Indique la persistance du message reçu. L'en-tête peut avoir l'une des valeurs suivantes:

#### **nonPersistent**

Le message ne survit pas aux défaillances du système ou aux redémarrages du gestionnaire de files d'attente.

#### **persistant**

Le message survit aux échecs du système ou aux redémarrages du gestionnaire de files d'attente.

#### **V 9.3.0** **REST API V3** **ibm-mq-md-priority**

Renvoie le paramètre de la priorité du message. Exemple :

```
ibm-mq-md-priority: 3
```

### **ibm-mq-md-replyTo**

Indique la destination de réponse pour le message reçu. Le format de l'en-tête utilise la notation standard de la file d'attente de réponse et du gestionnaire de files d'attente, `replyQueue@replyQmgr`.

Exemple :

```
ibm-mq-md-replyTo: myReplyQueue@myReplyQMGR
```

#### **V 9.3.3** **ibm-mq-resolved-qmgr**

Indique le nom du gestionnaire de files d'attente utilisé pour la demande. Si un nom unique a été utilisé dans la ressource URL, cet en-tête identifie le nom du gestionnaire de files d'attente associé à ce nom unique. Si le nom unique utilisé dans la ressource URL fait référence à un groupe de gestionnaires de files d'attente, cet en-tête identifie le gestionnaire de files d'attente utilisé dans le groupe.

#### **V 9.3.0** **REST API V3** **ibm-mq-usr**

Renvoie les propriétés définies par l'utilisateur du message. Plusieurs propriétés peuvent être définies sur un message, auquel cas il y aura au moins deux instances distinctes de l'en-tête de réponse `ibm-mq-usr`.

Exemple :

```
ibm-mq-usr: myIPprop;5;short
ibm-mq-usr: mySProp;"hi";string
ibm-mq-usr: myBProp>true;boolean
```

La syntaxe des propriétés est la suivante:

```
ibm-mq-usr: property_name; user_value; user_type
```

### **nom\_propriété**

Nom de la propriété utilisateur spécifiée. Il doit s'agir d'un nom de propriété JMS valide.

### **valeur\_utilisateur**

Valeur de la propriété.

### **type\_utilisateur**

Type de la propriété:

- boolean (true/false, MQBOOL)
- byte (entier 8 bits, MQINT8)
- short (entier 16 bits, MQINT16)
- integer (entier 32 bits, MQINT32)
- long (entier 64 bits, MQINT64)
- float (réel 32 bits, MQFLOAT32)
- double (réel 64 bits, MQFLOAT64)
- string (chaîne entre guillemets)

## **Format du corps de la réponse**

En cas de réussite, le corps de la réponse contient le corps du message reçu. Si une erreur se produit, le corps de la réponse contient un message d'erreur au format JSON. Les deux réponses sont codées par UTF-8. Pour plus d'informations, voir [Traitement des erreurs REST API](#).

Sachez que lors de la réception d'un message, seuls les messages formatés IBM MQ MQSTR ou JMS TextMessage sont pris en charge.

La consultation d'une file d'attente marquée comme GET interdite ne renvoie aucun contenu.

Si la file d'attente explorée contient des messages avec des identificateurs de message en double, le premier message est renvoyé lors du filtrage sur l'identificateur de message.

## **Exemples**

Les exemples ci-après utilisent l'URL de ressource de version 2. Si vous utilisez une version de IBM MQ antérieure à IBM MQ 9.1.5, vous devez utiliser l'URL de ressource v1 à la place. En d'autres cas, dans l'URL de ressource, remplacez v1 où l'exemple d'URL utilise v2.

L'exemple suivant permet de se connecter à un utilisateur appelé `muser` avec le mot de passe `muser`. Dans cURL, la demande de connexion peut ressembler à l'exemple Windows suivant. Le jeton LTPA est stocké dans le fichier `cookiejar.txt` à l'aide de l'indicateur `-c` :

```
curl -k "https://localhost:9443/ibmmq/rest/v2/login" -X POST
-H "Content-Type: application/json" --data "{\"username\":\"muser\", \"password\":\"muser\"}"
-c c:\cookiejar.txt
```

Une fois l'utilisateur connecté, le jeton LTPA et l'en-tête `ibm-mq-rest-csrf-token` HTTP sont utilisés pour authentifier d'autres demandes. Le `ibm-mq-rest-csrf-token` `token_value` peut être n'importe quelle valeur, y compris vide.



## URL de ressource

`https://host:port/ibmmq/rest/v1/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

`https://host:port/ibmmq/rest/v2/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

**V 9.3.0** `https://host:port/ibmmq/rest/v3/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

### qmgrName

Indique le nom du gestionnaire de files d'attente auquel se connecter pour la messagerie. Le gestionnaire de files d'attente doit se trouver sur la même machine que le serveur mqweb.

**V 9.3.3** Depuis IBM MQ 9.3.3, vous pouvez vous connecter à un gestionnaire de files d'attente local ou à un gestionnaire de files d'attente éloignées. Le nom que vous spécifiez pour **qmgrName** dépend de la manière dont votre serveur mqweb est configuré :

- Si votre serveur mqweb est configuré pour se connecter uniquement aux gestionnaires de files d'attente locaux, utilisez le nom du gestionnaire de files d'attente.
- Si votre serveur mqweb est configuré pour connecter des gestionnaires de files d'attente éloignées, utilisez le nom unique du gestionnaire de files d'attente comme indiqué dans les informations de connexion du gestionnaire de files d'attente éloignées.

Vous pouvez déterminer si votre serveur mqweb est configuré pour se connecter à des gestionnaires de files d'attente locaux ou à des gestionnaires de files d'attente éloignées à l'aide de la commande **dspmweb properties** et en affichant la propriété **mqRestMessagingConnectionMode**.

Le nom du gestionnaire de files d'attente est sensible à la casse.

Si le nom du gestionnaire de files d'attente inclut une barre oblique, un point ou le signe de pourcentage, ces caractères doivent être codés dans l'URL :

- Une barre oblique (/) doit être codée sous la forme %2F.
- Le signe pourcentage (%) doit être codé en tant que %25.

### queueName

Indique le nom de la file d'attente à partir de laquelle le message suivant doit être extrait.

La file d'attente doit être définie comme étant locale ou un alias pointant vers une file d'attente locale.

Le nom de la file d'attente est sensible à la casse.

Si le nom de la file d'attente contient une barre oblique ou un signe de pourcentage, les caractères suivants doivent être codés URL :

- Une barre oblique (/) doit être codée sous la forme %2F.
- Le signe de pourcentage, %, doit être codé en tant que %25.

Vous pouvez utiliser HTTP à la place de HTTPS si vous activez les connexions HTTP. Pour plus d'informations sur l'activation de HTTP, voir [Configuration des ports HTTP et HTTPS](#).

## Paramètres de requête facultatifs

**REST API V1** **REST API V2** **correlationId=hexValue**

Indique que la méthode HTTP renvoie le message suivant avec l'ID de corrélation correspondant.

### hexValue

Le paramètre de requête doit être indiqué sous la forme d'une chaîne codée hexadécimale de 48 caractères représentant 24 octets.

Exemple :

```
../message?correlationId=414d5120514d4144455620202020202067d8bf5923582e02
```

V 9.3.0

REST API V3

**correlationId= ID:hexValue** ou **correlationId=valeur\_spécifique\_application**

Indique que la méthode HTTP renvoie le message suivant avec l'ID de corrélation correspondant.

#### hexValue

Le paramètre de requête doit être indiqué sous la forme d'une chaîne codée hexadécimale de 48 caractères, représentant 24 octets, précédée de la chaîne "ID: ".

Exemple :

```
../message?correlationId=ID:414d5120514d4144455620202020202067d8bf5923582e02
```

#### valeur\_spécif\_application

Le paramètre de requête peut être spécifié en tant que chaîne spécifique à l'application.

Exemple :

```
../message?correlationId=My-Custom-CorrelId
```

REST API V1

REST API V2

**messageId=hexValue**

Indique que la méthode HTTP renvoie le message suivant avec l'ID de message correspondant.

#### hexValue

Le paramètre de requête doit être indiqué sous la forme d'une chaîne codée hexadécimale de 48 caractères représentant 24 octets.

Exemple :

```
../message?messageId=414d5120514d4144455620202020202067d8ce5923582f07
```

V 9.3.0

REST API V3

**messageId= ID:hexValue**

Indique que la méthode HTTP renvoie le message suivant avec l'ID de message correspondant.

#### hexValue

Le paramètre de requête doit être indiqué sous la forme d'une chaîne codée hexadécimale de 48 caractères, représentant 24 octets, précédée de la chaîne "ID: ".

Exemple :

```
../message?messageId=ID:414d5120514d4144455620202020202067d8ce5923582f07
```

#### wait=integerValue

Indique que la méthode HTTP attendra *integerValue* millisecondes pour que le prochain message soit disponible.

#### integerValue

Le paramètre de requête doit être spécifié sous la forme d'un entier représentant la durée en millisecondes. La valeur maximale est 2147483647.

Exemple :

```
../message?wait=120000
```

### En-têtes de demande :

Les en-têtes suivants doivent être envoyés avec la demande:

#### Autorisation

Cet en-tête doit être envoyé si vous utilisez l'authentification de base. Pour plus d'informations, voir [Using HTTP basic authentication with the REST API](#).

#### ibm-mq-rest-csrf-token

Cet en-tête doit être défini, mais sa valeur est indifférente, et peut notamment être vide.

Les en-têtes suivants peuvent éventuellement être envoyés avec la demande:

### Accepter-Ensemble de caractères

Cet en-tête peut être utilisé pour indiquer quel jeu de caractères est acceptable pour la réponse. S'il est spécifié, cet en-tête doit être défini sur UTF-8.

### Langage d'acceptation

Cet en-tête indique la langue requise pour les exceptions ou les messages d'erreur renvoyés dans le corps du message de réponse.

## Format du corps de la demande




Néant.

## Sécurité


L'appelant doit être authentifié auprès du serveur mqweb. Les rôles MQWebAdmin et MQWebAdminRO ne sont pas applicables pour le messaging REST API. Pour plus d'informations sur la sécurité pour REST API, voir [Sécurité IBM MQ Console et REST API](#).

Une fois authentifié auprès du serveur mqweb, l'utilisateur est capable d'utiliser à la fois messaging REST API et administrative REST API.

Le principal de sécurité de l'appelant doit avoir la possibilité d'extraire des messages de la file d'attente spécifiée:

- La file d'attente spécifiée par la partie *{queueName}* de l'URL de ressource doit être activée pour GET.
-   Pour la file d'attente spécifiée par la partie *{queueName}* de l'URL de ressource, les droits +GET, +INQ et +BROWSE doivent être accordés au principal de sécurité de l'appelant.
-  Pour la file d'attente spécifiée par la partie *{queueName}* de l'URL de ressource, UPDATE, l'accès doit être accordé au principal de sécurité de l'appelant.

Ce principal de sécurité peut être l'utilisateur qui a démarré le serveur mqweb ou l'utilisateur qui est connecté au serveur mqweb. Si le gestionnaire de files d'attente auquel vous vous connectez est un gestionnaire de files d'attente éloignées, le principal de sécurité peut être l'utilisateur fourni par les données d'identification dans les informations de connexion du gestionnaire de files d'attente éloignées ou un autre utilisateur déterminé par les règles de sécurité du canal. Pour plus d'informations, voir [Détermination du principal de sécurité utilisé par messaging REST API](#).

 Sous AIX, Linux, and Windows, vous pouvez accorder des droits aux principaux de sécurité pour utiliser les ressources IBM MQ à l'aide de la commande **setmqaut**. Pour plus d'informations, voir [setmqaut \(grant of revoke authority\)](#).

 Sous z/OS, voir [Configuration de la sécurité sous z/OS](#).

## Codes de statut de réponse

### 200

Le message a été reçu.

### 204

Aucun message disponible.

### 400

Des données non valides ont été fournies.

Par exemple, une valeur de paramètre de requête non valide a été spécifiée.

### 401

Non authentifié.

L'appelant doit être authentifié sur le serveur mqweb et doit être membre d'un ou plusieurs rôles MQWebAdmin, MQWebAdminRO ou MQWebUser. L'en-tête `ibm-mq-rest-csrf-token` doit également être spécifié. Pour plus d'informations, voir «Sécurité», à la page 2279.

#### 403

Non autorisé

L'appelant est authentifié auprès du serveur mqweb et est associé à un principal valide. Toutefois, le principal n'a pas accès à l'ensemble ou à un sous-ensemble des ressources IBM MQ requises ou n'est pas dans le rôle MQWebUser. Pour plus d'informations sur l'accès requis, voir «Sécurité», à la page 2279.

#### 404

La file d'attente n'existe pas.

#### 405

L'opération GET est interdite pour la file d'attente.

#### 500

Problème de serveur ou code d'erreur provenant de IBM MQ.

#### 501

La réponse HTTP n'a pas pu être construite.

Par exemple, le type du message reçu est incorrect ou correct, mais le corps n'a pas pu être traité.

#### 502

Le principal de sécurité en cours ne peut pas recevoir le message car le fournisseur de messagerie ne prend pas en charge la fonction requise. Par exemple, si le chemin d'accès aux classes du serveur mqweb n'est pas valide.

#### 503

Gestionnaire de files d'attente non en cours d'exécution.

### En-têtes de réponse :

Les en-têtes suivants sont renvoyés avec la réponse:

#### Contenu-Langue

Indique l'identificateur de langue du message de réponse en cas d'erreurs ou d'exceptions. Utilisé conjointement avec l'en-tête de demande `Accept-Language` pour indiquer la langue requise pour toute condition d'erreur ou d'exception. La valeur par défaut du serveur mqweb est utilisée si la langue demandée n'est pas prise en charge.

#### Longueur de contenu

Indique la longueur du corps de la réponse HTTP, même en l'absence de contenu. La valeur contient la longueur (octets) des données de message.

#### Content-Type

Indique le type de contenu renvoyé dans le corps de réponse du message reçu. En cas de réussite, la valeur est `text/plain; charset=utf-8`. En cas d'erreurs ou d'exceptions, la valeur est `application/json; charset=utf-8`.

#### REST API V1 > REST API V2 **ibm-mq-md-correlationId**

Indique l'ID de corrélation du message reçu. L'en-tête est renvoyé si le message reçu contient un ID de corrélation valide. Elle est représentée sous la forme d'une chaîne codée hexadécimale de 48 caractères, représentant 24 octets.

Exemple :

```
ibm-mq-md-correlationId: 414d5120514d41444556202020202067d8bf5923582e02
```

#### V 9.3.0 > REST API V3 **ibm-mq-md-correlationId**

Indique l'ID de corrélation du message reçu. L'en-tête est renvoyé si le message reçu contient un ID de corrélation valide. L'ID corrélation peut prendre l'une des formes suivantes:



- Chaîne codée en hexadécimal de 48 caractères, représentant 24 octets, précédée de la chaîne "ID:". Exemple :

```
ibm-mq-md-correlationId: ID:414d5120514d4144455620202020202067d8bf5923582e02
```

- Valeur spécifique à l'application. La valeur est une chaîne spécifique à l'application:

```
ibm-mq-md-correlationId: My-Custom-CorrelId
```

### **ibm-mq-md-expiration**

Indique la durée d'expiration restante du message reçu. L'en-tête peut avoir l'une des valeurs suivantes:

#### **illimité**

Le message n'expire pas.

#### **Valeur entière**

Nombre de millisecondes restant avant l'expiration du message.

### **REST API V1** **REST API V2** **ibm-mq-md-messageId**

Indique l'ID de message attribué par IBM MQ à ce message. Tout comme l'en-tête `ibm-mq-md-correlationId`, il est représenté sous la forme d'une chaîne codée hexadécimale de 48 caractères représentant 24 octets.

Exemple :

```
ibm-mq-md-messageId: 414d5120514d4144455620202020202067d8ce5923582f07
```

### **V 9.3.0** **REST API V3** **ibm-mq-md-messageId**

Indique l'ID de message attribué par IBM MQ à ce message. Comme l'en-tête `ibm-mq-md-correlationId`, il est représenté sous la forme d'une chaîne codée hexadécimale de 48 caractères, représentant 24 octets, préfixée avec la chaîne "ID:"

Exemple :

```
ibm-mq-md-messageId: ID:414d5120514d4144455620202020202067d8ce5923582f07
```

### **ibm-mq-md-persistence**

Indique la persistance du message reçu. L'en-tête peut avoir l'une des valeurs suivantes:

#### **nonPersistent**

Le message ne survit pas aux défaillances du système ou aux redémarrages du gestionnaire de files d'attente.

#### **persistant**

Le message survit aux échecs du système ou aux redémarrages du gestionnaire de files d'attente.

### **V 9.3.0** **REST API V3** **ibm-mq-md-priority**

Renvoie le paramètre de la priorité du message. Exemple :

```
ibm-mq-md-priority: 3
```

### **ibm-mq-md-replyTo**

Indique la destination de réponse pour le message reçu. Le format de l'en-tête utilise la notation standard de la file d'attente de réponse et du gestionnaire de files d'attente, `replyQueue@replyQmgr`.

Exemple :

```
ibm-mq-md-replyTo: myReplyQueue@myReplyQMGR
```

### V 9.3.3 **ibm-mq-resolved-qmgr**

Indique le nom du gestionnaire de files d'attente utilisé pour la demande. Si un nom unique a été utilisé dans la ressource URL, cet en-tête identifie le nom du gestionnaire de files d'attente associé à ce nom unique. Si le nom unique utilisé dans la ressource URL fait référence à un groupe de gestionnaires de files d'attente, cet en-tête identifie le gestionnaire de files d'attente utilisé dans le groupe.

### V 9.3.0 **REST API V3** **ibm-mq-usr**

Renvoie les propriétés définies par l'utilisateur du message. Plusieurs propriétés peuvent être définies sur un message, auquel cas il y aura au moins deux instances distinctes de l'en-tête de réponse `ibm-mq-usr`.

Exemple :

```
ibm-mq-usr: myIPprop;5;short
ibm-mq-usr: mySProp;"hi";string
ibm-mq-usr: myBProp>true;boolean
```

La syntaxe des propriétés est la suivante:

```
ibm-mq-usr: property_name; user_value; user_type
```

#### **nom\_propriété**

Nom de la propriété utilisateur spécifiée. Il doit s'agir d'un nom de propriété JMS valide.

#### **valeur\_utilisateur**

Valeur de la propriété.

#### **type\_utilisateur**

Type de la propriété:

- `boolean` (true/false, MQBOOL)
- `byte` (entier 8 bits, MQINT8)
- `short` (entier 16 bits, MQINT16)
- `integer` (entier 32 bits, MQINT32)
- `long` (entier 64 bits, MQINT64)
- `float` (réel 32 bits, MQFLOAT32)
- `double` (réel 64 bits, MQFLOAT64)
- `string` (chaîne entre guillemets)

## **Format du corps de la réponse**

En cas de réussite, le corps de la réponse contient le corps du message reçu. Si une erreur se produit, le corps de la réponse contient un message d'erreur au format JSON. Les deux réponses sont codées par UTF-8. Pour plus d'informations, voir [Traitement des erreurs REST API](#).

Sachez que lors de la réception d'un message, seuls les messages au format IBM MQ MQSTR et JMS `TextMessage` sont pris en charge. Par la suite, tous les messages sont reçus sous le point de synchronisation et tous les messages non gérés sont laissés dans la file d'attente. La file d'attente IBM MQ peut être configurée pour déplacer ces messages incohérents vers une autre destination. Pour plus d'informations, voir [Gestion des messages incohérents dans IBM MQ classes for JMS](#).

## **Exemples**

Les exemples ci-après utilisent l'URL de ressource de version 2. Si vous utilisez une version de IBM MQ antérieure à IBM MQ 9.1.5, vous devez utiliser l'URL de ressource v1 à la place. En d'autres cas, dans l'URL de ressource, remplacez v1 où l'exemple d'URL utilise v2.



- «Sécurité», à la page 2286
- «Codes de statut de réponse», à la page 2287
- «En-têtes de réponse :», à la page 2287
- «Format du corps de la réponse», à la page 2288
- «Exemples», à la page 2288

## URL de ressource

`https://host:port/ibmmq/rest/v1/messaging/qmgr/{qmgrName}/queue/{queueName}/messagelist`

`https://host:port/ibmmq/rest/v2/messaging/qmgr/{qmgrName}/queue/{queueName}/messagelist`

**V 9.3.0** `https://host:port/ibmmq/rest/v3/messaging/qmgr/{qmgrName}/queue/{queueName}/messagelist`

### qmgrName

Indique le nom du gestionnaire de files d'attente auquel se connecter pour la messagerie. Le gestionnaire de files d'attente doit se trouver sur la même machine que le serveur mqweb.

**V 9.3.3** Depuis IBM MQ 9.3.3, vous pouvez vous connecter à un gestionnaire de files d'attente local ou à un gestionnaire de files d'attente éloignées. Le nom que vous spécifiez pour **qmgrName** dépend de la manière dont votre serveur mqweb est configuré:

- Si votre serveur mqweb est configuré pour se connecter uniquement aux gestionnaires de files d'attente locaux, utilisez le nom du gestionnaire de files d'attente.
- Si votre serveur mqweb est configuré pour connecter des gestionnaires de files d'attente éloignées, utilisez le nom unique du gestionnaire de files d'attente comme indiqué dans les informations de connexion du gestionnaire de files d'attente éloignées.

Vous pouvez déterminer si votre serveur mqweb est configuré pour se connecter à des gestionnaires de files d'attente locaux ou à des gestionnaires de files d'attente éloignées à l'aide de la commande **dspmweb properties** et en affichant la propriété **mqRestMessagingConnectionMode**.

Le nom du gestionnaire de files d'attente est sensible à la casse.

Si le nom du gestionnaire de files d'attente inclut une barre oblique, un point ou le signe de pourcentage, ces caractères doivent être codés dans l'URL :

- Une barre oblique (/) doit être codée sous la forme %2F.
- Le signe pourcentage (%) doit être codé en tant que %25.

### queueName

Indique le nom de la file d'attente à partir de laquelle parcourir les messages.

La file d'attente doit être définie comme étant locale ou un alias pointant vers une file d'attente locale.

Le nom de la file d'attente est sensible à la casse.

Si le nom de la file d'attente contient une barre oblique ou un signe de pourcentage, les caractères suivants doivent être codés URL :

- Une barre oblique (/) doit être codée sous la forme %2F.
- Le signe de pourcentage, %, doit être codé en tant que %25.

Vous pouvez utiliser HTTP à la place de HTTPS si vous activez les connexions HTTP . Pour plus d'informations sur l'activation de HTTP, voir [Configuration des ports HTTP et HTTPS](#).

## Paramètres de requête facultatifs

### REST API V2 **correlationId=hexValue**

Indique que la méthode HTTP renvoie le message suivant avec l'ID de corrélation correspondant.

#### hexValue

Le paramètre de requête doit être indiqué sous la forme d'une chaîne codée hexadécimale de 48 caractères représentant 24 octets.

Exemple :

```
../messagelist?correlationId=414d5120514d41444556202020202067d8bf5923582e02
```

### V 9.3.0 REST API V3 **correlationId= ID:hexValue ou correlationId=valeur\_spécifique\_application**

Indique que la méthode HTTP renvoie une liste de messages avec l'ID de corrélation correspondant.

#### hexValue

Le paramètre de requête doit être indiqué sous la forme d'une chaîne codée hexadécimale de 48 caractères, représentant 24 octets, précédée de la chaîne "ID : ".

Exemple :

```
../message?correlationId=ID:414d5120514d41444556202020202067d8bf5923582e02
```

#### valeur\_spécif\_application

Le paramètre de requête peut être spécifié en tant que chaîne spécifique à l'application.

Exemple :

```
../message?correlationId=My-Custom-CorrelId
```

### REST API V1 REST API V2 **messageId=hexValue**

Indique que la méthode HTTP renvoie le message suivant avec l'ID de message correspondant.

#### hexValue

Le paramètre de requête doit être indiqué sous la forme d'une chaîne codée hexadécimale de 48 caractères représentant 24 octets.

Exemple :

```
../message?messageId=414d5120514d41444556202020202067d8ce5923582f07
```

### V 9.3.0 REST API V3 **messageId= ID:hexValue**

Indique que la méthode HTTP renvoie le message suivant avec l'ID de message correspondant.

#### hexValue

Le paramètre de requête doit être indiqué sous la forme d'une chaîne codée hexadécimale de 48 caractères, représentant 24 octets, précédée de la chaîne "ID : ".

Exemple :

```
../message?messageId=ID:414d5120514d41444556202020202067d8ce5923582f07
```

### **limit=integerValue**

Indique que le corps de réponse de la méthode HTTP est limité aux éléments JSON *integerValue* .

#### integerValue

Le paramètre de requête doit être spécifié sous la forme d'une valeur entière qui représente le nombre maximal d'éléments contenus dans le corps de réponse JSON.

La valeur par défaut est 10 et la valeur maximale est 2147483647.

Exemple :

```
../messageList?limit=250
```

## En-têtes de demande :

Les en-têtes suivants doivent être envoyés avec la demande:

### Autorisation

Cet en-tête doit être envoyé si vous utilisez l'authentification de base. Pour plus d'informations, voir [Using HTTP basic authentication with the REST API](#).

### ibm-mq-rest-csrf-token

Cet en-tête doit être défini, mais sa valeur est indifférente, et peut notamment être vide.

Les en-têtes suivants peuvent éventuellement être envoyés avec la demande:

### Accepter-Ensemble de caractères

Cet en-tête peut être utilisé pour indiquer quel jeu de caractères est acceptable pour la réponse. S'il est spécifié, cet en-tête doit être défini sur UTF-8.

### Langage d'acceptation

Cet en-tête indique la langue requise pour les exceptions ou les messages d'erreur renvoyés dans le corps du message de réponse.

## Format du corps de la demande




Néant.

## Sécurité


L'appelant doit être authentifié auprès du serveur mqweb. Les rôles MQWebAdmin et MQWebAdminRO ne sont pas applicables pour le messaging REST API. Pour plus d'informations sur la sécurité pour REST API, voir [Sécurité IBM MQ Console et REST API](#).

Une fois authentifié auprès du serveur mqweb, l'utilisateur est capable d'utiliser à la fois messaging REST API et administrative REST API.

Le principal de sécurité de l'appelant doit avoir le droit de parcourir les messages dans la file d'attente spécifiée :

- La file d'attente spécifiée par la partie *{queueName}* de l'URL de ressource doit être activée pour BROWSE.
-   Pour la file d'attente spécifiée par la partie *{queueName}* de l'URL de ressource, les droits +GET, +INQ et +BROWSE doivent être accordés au principal de sécurité de l'appelant.
-  Pour la file d'attente spécifiée par la partie *{queueName}* de l'URL de ressource, UPDATE, l'accès doit être accordé au principal de sécurité de l'appelant.

Ce principal de sécurité peut être l'utilisateur qui a démarré le serveur mqweb ou l'utilisateur qui est connecté au serveur mqweb. Si le gestionnaire de files d'attente auquel vous vous connectez est un gestionnaire de files d'attente éloignées, le principal de sécurité peut être l'utilisateur fourni par les données d'identification dans les informations de connexion du gestionnaire de files d'attente éloignées ou un autre utilisateur déterminé par les règles de sécurité du canal. Pour plus d'informations, voir [Détermination du principal de sécurité utilisé par messaging REST API](#).

 Sous AIX, Linux, and Windows, vous pouvez accorder des droits aux principaux de sécurité pour utiliser les ressources IBM MQ à l'aide de la commande **setmqaut**. Pour plus d'informations, voir [setmqaut \(grant of revoke authority\)](#).

 Sous z/OS, voir [Configuration de la sécurité sous z/OS](#).

## Codes de statut de réponse

### 200

La réception de la liste de messages a abouti.

### 400

Des données non valides ont été fournies.

Par exemple, une valeur de paramètre de requête non valide a été spécifiée.

### 401

Non authentifié.

L'appelant doit être authentifié sur le serveur mqweb et doit être membre d'un ou plusieurs rôles MQWebAdmin, MQWebAdminRO ou MQWebUser. L'en-tête `ibm-mq-rest-csrf-token` doit également être spécifié. Pour plus d'informations, voir «Sécurité», à la page 2286.

### 403

Non autorisé

L'appelant est authentifié auprès du serveur mqweb et est associé à un principal valide. Toutefois, le principal n'a pas accès à l'ensemble ou à un sous-ensemble des ressources IBM MQ requises ou n'est pas dans le rôle MQWebUser. Pour plus d'informations sur l'accès requis, voir «Sécurité», à la page 2286.

### 404

La file d'attente n'existe pas.

### 500

Problème de serveur ou code d'erreur provenant de IBM MQ.

### 501

La réponse HTTP n'a pas pu être construite.

Par exemple, le type du message reçu est incorrect ou correct, mais le corps n'a pas pu être traité.

### 502

Le principal de sécurité en cours ne peut pas recevoir le message car le fournisseur de messagerie ne prend pas en charge la fonction requise. Par exemple, si le chemin d'accès aux classes du serveur mqweb n'est pas valide.

### 503

Gestionnaire de files d'attente non en cours d'exécution.

## En-têtes de réponse :

### Contenu-Langue

Indique l'identificateur de langue du message de réponse en cas d'erreurs ou d'exceptions. Utilisé conjointement avec l'en-tête de demande `Accept-Language` pour indiquer la langue requise pour toute condition d'erreur ou d'exception. La valeur par défaut du serveur mqweb est utilisée si la langue demandée n'est pas prise en charge.

### Longueur de contenu

Indique la longueur du corps de la réponse HTTP, même en l'absence de contenu. La valeur contient la longueur des données du message, en octets.

### Content-Type

Indique le type de corps de réponse. La valeur est `application/json; charset=utf-8`.

### **V 9.3.3** `ibm-mq-resolved-qmgr`

Indique le nom du gestionnaire de files d'attente utilisé pour la demande. Si un nom unique a été utilisé dans la ressource URL, cet en-tête identifie le nom du gestionnaire de files d'attente associé à ce nom unique. Si le nom unique utilisé dans la ressource URL fait référence à un groupe de gestionnaires de files d'attente, cet en-tête identifie le gestionnaire de files d'attente utilisé dans le groupe.

## Format du corps de la réponse

En cas de réussite, le corps de la réponse est une réponse codée UTF-8 . La réponse contient un objet JSON externe qui contient un tableau JSON unique appelé `messages`. Chaque élément du tableau est un objet JSON qui contient des informations sur un message de la file d'attente. Chaque élément contient les attributs suivants:

### REST API V1 > REST API V2 `correlationId`

Indique l'ID de corrélation du message. La valeur est renvoyée si le message contient un ID de corrélation valide.

### V 9.3.0 > REST API V3 `correlationId`

Indique l'ID de corrélation du message. La valeur est renvoyée si le message contient un ID de corrélation valide. L'ID de corrélation est précédé de la chaîne "ID:" ou peut être une valeur spécifique à l'application.

### REST API V1 > REST API V2 `messageId`

Indique l'ID de message alloué par IBM MQ à ce message. Elle est représentée sous la forme d'une chaîne codée hexadécimale de 48 caractères représentant 24 octets.

### V 9.3.0 > REST API V3 `messageId`

Indique l'ID de message alloué par IBM MQ à ce message. Elle est représentée sous la forme d'une chaîne codée hexadécimale de 48 caractères représentant 24 octets. L'ID message est précédé de la chaîne "ID:".

### format

Indique la zone de format MQMD. Dans des circonstances normales, les messages texte contiennent la valeur IBM MQ MQSTR .

Si une demande est effectuée pour obtenir une liste de messages dans une file d'attente pour laquelle l'opération GET est interdite, un tableau JSON vide est renvoyé.

Si une erreur se produit, le corps de la réponse contient un message d'erreur au format JSON. Pour plus d'informations, voir [Traitement des erreurs REST API](#).

## Exemples

Les exemples ci-après utilisent l'URL de ressource de version 2. Si vous utilisez une version de IBM MQ antérieure à IBM MQ 9.1.5, vous devez utiliser l'URL de ressource v1 à la place. En d'autres cas, dans l'URL de ressource, remplacez v1 où l'exemple d'URL utilise v2.

L'exemple suivant permet de se connecter à un utilisateur appelé `muser` avec le mot de passe `muser`. Dans cURL, la demande de connexion peut ressembler à l'exemple Windows suivant. Le jeton LTPA est stocké dans le fichier `cookiejar.txt` à l'aide de l'indicateur `-c` :

```
curl -k "https://localhost:9443/ibmmq/rest/v2/login" -X POST
-H "Content-Type: application/json" --data "{\"username\":\"muser\",\"password\":\"muser\"}"
-c c:\cookiejar.txt
```

Une fois l'utilisateur connecté, le jeton LTPA et l'en-tête `ibm-mq-rest-csrf-token` HTTP sont utilisés pour authentifier d'autres demandes. Le `ibm-mq-rest-csrf-token` `token_value` peut être n'importe quelle valeur, y compris vide.

- L'exemple Windows cURL suivant répertorie les dix messages disponibles suivants de la file d'attente Q1 sur le gestionnaire de files d'attente QM1, à l'aide des options par défaut:

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/Q1/messageList"
-X GET -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token-value"
-H "Accept: application/json"
```





- «Exemples», à la page 2294

## URL de ressource

`https://host:port/ibmmq/rest/v2/messaging/qmgr/{qmgrName}/topic/{topicString}/message`

**V 9.3.0** `https://host:port/ibmmq/rest/v3/messaging/qmgr/{qmgrName}/topic/{topicString}/message`

### qmgrName

Indique le nom d'un gestionnaire de files d'attente auquel se connecter pour la messagerie.

**V 9.3.3** Depuis IBM MQ 9.3.3, vous pouvez vous connecter à un gestionnaire de files d'attente local ou à un gestionnaire de files d'attente éloignées. Le nom que vous spécifiez pour **qmgrName** dépend de la manière dont votre serveur mqweb est configuré:

- Si votre serveur mqweb est configuré pour se connecter uniquement aux gestionnaires de files d'attente locaux, utilisez le nom du gestionnaire de files d'attente.
- Si votre serveur mqweb est configuré pour connecter des gestionnaires de files d'attente éloignées, utilisez le nom unique du gestionnaire de files d'attente comme indiqué dans les informations de connexion du gestionnaire de files d'attente éloignées.

Vous pouvez déterminer si votre serveur mqweb est configuré pour se connecter à des gestionnaires de files d'attente locaux ou à des gestionnaires de files d'attente éloignées à l'aide de la commande **dspmweb properties** et en affichant la propriété **mqRestMessagingConnectionMode**.

Le nom est sensible à la casse.

Si le nom inclut une barre oblique, un point ou un signe de pourcentage, les caractères suivants doivent être codés dans l'URL :

- Une barre oblique doit être codée en tant que %2F.
- Un point doit être codé en tant que %2E.
- Un signe de pourcentage doit être codé en tant que %25.

### topicString

Indique la chaîne de rubrique sur laquelle le message doit être publié.

La chaîne de rubrique est sensible à la casse. La chaîne de rubrique peut contenir plusieurs niveaux de rubrique, séparés par une barre oblique.

Si la chaîne de rubrique contient un signe de pourcentage, un point ou un point d'interrogation, les caractères suivants doivent être codés dans l'URL :

- Un signe de pourcentage doit être codé en tant que %25.
- Un point doit être codé en tant que %2E.
- Un point d'interrogation doit être codé en tant que %3F.

Si la chaîne de rubrique commence ou se termine par une barre oblique, elle doit être codée avec une %2F.

Par exemple, pour publier dans la chaîne de rubrique:

- `sport/football` sur le gestionnaire de files d'attente `MY.QMGR`, vous utilisez l'URL suivante:

```
https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/MY%2EQMGR/topic/sport/football/message
```

- `/sport/football` sur le gestionnaire de files d'attente `MY.QMGR`, vous utilisez l'URL suivante:

```
https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/MY%2EQMGR/topic/%2Fsport/football/message
```

Vous pouvez utiliser HTTP à la place de HTTPS si vous activez les connexions HTTP. Pour plus d'informations sur l'activation de HTTP, voir [Configuration des ports HTTP et HTTPS](#).

## En-têtes de demande :

Les en-têtes suivants doivent être envoyés avec la demande:

### Autorisation

Cet en-tête doit être envoyé si vous utilisez l'authentification de base. Pour plus d'informations, voir [Using HTTP basic authentication with the REST API](#).

### Content-Type

Cet en-tête doit être envoyé avec l'une des valeurs suivantes:

- text/plain;charset=utf-8
- text/html;charset=utf-8
- text/xml;charset=utf-8
- application/json;charset=utf-8
- application/xml;charset=utf-8

### ibm-mq-rest-csrf-token

Cet en-tête doit être défini, mais sa valeur est indifférente, et peut notamment être vide.

Les en-têtes suivants peuvent éventuellement être envoyés avec la demande:

### Langage d'acceptation

Cet en-tête indique la langue requise pour les exceptions ou les messages d'erreur renvoyés dans le corps du message de réponse.

### ibm-mq-md-expiration

Cet en-tête définit la durée d'expiration du message créé. L'expiration d'un message commence à partir du moment où le message arrive sur le gestionnaire de files d'attente. Par conséquent, le temps d'attente du réseau est ignoré. L'en-tête doit être spécifié sous la forme de l'une des valeurs suivantes:

#### illimité

Le message n'expire pas.

Cette valeur est la valeur par défaut.

#### Valeur entière

Millisecondes avant l'expiration du message.

Limité à la plage de 0 à 99999999900.

### ibm-mq-md-persistence

Cet en-tête définit la persistance du message créé. L'en-tête doit être spécifié sous la forme de l'une des valeurs suivantes:

#### nonPersistent

Le message ne survit pas aux défaillances du système ou aux redémarrages du gestionnaire de files d'attente.

Cette valeur est la valeur par défaut.

#### persistant

Le message survit aux échecs du système ou aux redémarrages du gestionnaire de files d'attente.

V 9.3.0

REST API V3

### ibm-mq-md-priority

Cet en-tête définit la priorité du message créé. L'en-tête doit être spécifié sous la forme de l'une des valeurs suivantes:

#### asDestination

Le message utilise la priorité spécifiée dans l'attribut DEFPRTY de l'objet de file d'attente IBM MQ sous-jacent.

#### Valeur entière

Indiquez la priorité réelle sous la forme d'un entier compris entre 0 et 9.

Exemple :

```
ibm-mq-md-priority: asDestination
```

### **ibm-mq-md-replyTo**

Cet en-tête définit la destination de réponse pour le message créé. Le format de l'en-tête utilise la notation standard consistant à fournir la file d'attente de réponse et un gestionnaire de files d'attente facultatif: `replyQueue[@replyQmgr]`

Exemple :

```
ibm-mq-md-replyTo: myReplyQueue@myReplyQMGR
```

### **V 9.3.0 REST API V3 ibm-mq-usr**

Définissez les propriétés définies par l'utilisateur du message de demande. Plusieurs propriétés peuvent être définies sur un message. Vous pouvez spécifier plusieurs propriétés séparées par des virgules dans un même en-tête de demande `ibm-mq-usr` ou utiliser deux ou plusieurs instances distinctes de l'en-tête de demande `ibm-mq-usr`.

Exemple :

```
ibm-mq-usr: myIPprop;5;short
ibm-mq-usr: mySProp;"hi";string
ibm-mq-usr: myBProp;true;boolean
ibm-mq-usr: myA;5;byte,myB;-10;integer
```

La syntaxe des propriétés est la suivante:

```
ibm-mq-usr: property_name; user_value; user_type
```

#### **nom\_propriété**

Nom de la propriété utilisateur spécifiée. Il doit s'agir d'un nom de propriété JMS valide.

#### **valeur\_utilisateur**

Valeur de la propriété.

#### **type\_utilisateur**

Type de la propriété:

- `boolean` (true/false, MQBOOL)
- `byte` (entier 8 bits, MQINT8)
- `short` (entier 16 bits, MQINT16)
- `integer` (entier 32 bits, MQINT32)
- `long` (entier 64 bits, MQINT64)
- `float` (réel 32 bits, MQFLOAT32)
- `double` (réel 64 bits, MQFLOAT64)
- `string` (chaîne entre guillemets)

## **Format du corps de la demande**

Le corps de la demande doit être du texte et utiliser le codage UTF-8 . Aucune structure de texte spécifique n'est requise. Un message au format MQSTR contenant le texte du corps de la demande est créé et publié dans la rubrique spécifiée.

**V 9.3.0 REST API V3** Si les propriétés définies par l'utilisateur de l'API REST V3 ou les fonctions d'ID de corrélation spécifiques à l'application sont utilisées, un message au format JMS `TextMessage` contenant le texte du corps de la demande est créé et placé dans la file d'attente spécifiée.




Pour plus d'informations, voir [Exemples](#).

## Sécurité


L'appelant doit être authentifié auprès du serveur mqweb. Les rôles MQWebAdmin et MQWebAdminRO ne sont pas applicables pour le messaging REST API. Pour plus d'informations sur la sécurité pour REST API, voir [Sécurité IBM MQ Console et REST API](#).

Une fois authentifié auprès du serveur mqweb, l'utilisateur est capable d'utiliser à la fois messaging REST API et administrative REST API.

Le principal de sécurité de l'appelant doit être autorisé à publier des messages dans la rubrique spécifiée:

- La rubrique spécifiée par la partie *{topicString}* de la ressource URL doit être PUBLISH activée.
-   Pour la rubrique spécifiée par la partie *{topicString}* de la ressource URL, les droits +PUB doivent être accordés au principal de sécurité de l'appelant.
-  Pour la rubrique spécifiée par la partie *{topicString}* de la ressource URL, l'accès UPDATE doit être accordé au principal de sécurité de l'appelant.

Ce principal de sécurité peut être l'utilisateur qui a démarré le serveur mqweb ou l'utilisateur qui est connecté au serveur mqweb. Si le gestionnaire de files d'attente auquel vous vous connectez est un gestionnaire de files d'attente éloignées, le principal de sécurité peut être l'utilisateur fourni par les données d'identification dans les informations de connexion du gestionnaire de files d'attente éloignées ou un autre utilisateur déterminé par les règles de sécurité du canal. Pour plus d'informations, voir [Détermination du principal de sécurité utilisé par messaging REST API](#).

 Sous AIX, Linux, and Windows, vous pouvez accorder des droits aux principaux de sécurité pour utiliser les ressources IBM MQ à l'aide de la commande **setmqaut**. Pour plus d'informations, voir [setmqaut](#) (grant of revoke authority).

 Sous z/OS, voir [Configuration de la sécurité sous z/OS](#).

Si vous utilisez Advanced Message Security (AMS) avec messaging REST API, notez que tous les messages sont chiffrés à l'aide du contexte du serveur mqweb et non du contexte de l'utilisateur qui a envoyé le message.

## Codes de statut de réponse

### 201

Le message a été créé et publié.

### 400

Des données non valides ont été fournies.

Par exemple, une valeur d'en-tête de demande non valide a été spécifiée.

### 401

Non authentifié.

L'appelant doit être authentifié sur le serveur mqweb et doit être membre d'un ou plusieurs rôles MQWebAdmin, MQWebAdminRO ou MQWebUser. L'en-tête `ibm-mq-rest-csrf-token` doit également être spécifié. Pour plus d'informations, voir «Sécurité», à la page 2293.

### 403

Non autorisé

L'appelant est authentifié auprès du serveur mqweb et est associé à un principal valide. Toutefois, le principal n'a pas accès à l'ensemble ou à un sous-ensemble des ressources IBM MQ requises ou n'est pas dans le rôle MQWebUser. Pour plus d'informations sur l'accès requis, voir «Sécurité», à la page 2293.

### 404

Le gestionnaire de files d'attente n'existe pas.

### 405

La rubrique est PUBLISH est interdite.

**415**

Un en-tête ou un corps de message est un type de support non pris en charge.

Par exemple, l'en-tête Content-Type est défini sur un type de support non pris en charge.

**500**

Problème de serveur ou code d'erreur provenant de IBM MQ.

**502**

Le principal de sécurité en cours ne peut pas publier le message car le fournisseur de messagerie ne prend pas en charge la fonction requise. Par exemple, si le chemin d'accès aux classes du serveur mqweb n'est pas valide.

**503**

Gestionnaire de files d'attente non en cours d'exécution.

**En-têtes de réponse :**

Les en-têtes suivants sont renvoyés avec la réponse:

**Contenu-Langue**

Indique l'identificateur de langue du message de réponse en cas d'erreurs ou d'exceptions. Utilisé conjointement avec l'en-tête de demande Accept-Language pour indiquer la langue requise pour toute condition d'erreur ou d'exception. La valeur par défaut du serveur mqweb est utilisée si la langue demandée n'est pas prise en charge.

**Longueur de contenu**

Indique la longueur du corps de la réponse HTTP, même en l'absence de contenu. En cas de réussite, la valeur est égale à zéro.

**Content-Type**

Indique le type de corps de réponse. En cas de réussite, la valeur est `text/plain; charset=utf-8`. En cas d'erreurs ou d'exceptions, la valeur est `application/json; charset=utf-8`.

**V 9.3.3 ibm-mq-resolved-qmgr**

Indique le nom du gestionnaire de files d'attente utilisé pour la demande. Si un nom unique a été utilisé dans la ressource URL, cet en-tête identifie le nom du gestionnaire de files d'attente associé à ce nom unique. Si le nom unique utilisé dans la ressource URL fait référence à un groupe de gestionnaires de files d'attente, cet en-tête identifie le gestionnaire de files d'attente utilisé dans le groupe.

**Format du corps de la réponse**

Le corps de la réponse est vide si le message est publié correctement. Si une erreur se produit, le corps de la réponse contient un message d'erreur. Pour plus d'informations, voir [Traitement des erreurs REST API](#).

**Exemples**

L'exemple suivant permet de se connecter à un utilisateur appelé `mquser` avec le mot de passe `mquser`. Dans cURL, la demande de connexion peut ressembler à l'exemple Windows suivant. Le jeton LTPA est stocké dans le fichier `cookiejar.txt` à l'aide de l'indicateur `-c` :

```
curl -k "https://localhost:9443/ibmmq/rest/v1/login" -X POST
-H "Content-Type: application/json" --data "{\"username\":\"mquser\", \"password\":\"mquser\"}"
-c c:\cookiejar.txt
```

Une fois l'utilisateur connecté, le jeton LTPA et l'en-tête `ibm-mq-rest-csrf-token` HTTP sont utilisés pour authentifier d'autres demandes. Le `ibm-mq-rest-csrf-token` `token_value` peut être n'importe quelle valeur, y compris vide.

- L'exemple Windows cURL suivant publie un message dans la chaîne de rubrique myTopic sur le gestionnaire de files d'attente QM1, à l'aide des options par défaut. Le message contient le texte *"Hello World !"*:

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/topic/myTopic/message"
-X POST -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token_value"
-H "Content-Type: text/plain;charset=utf-8" --data "Hello World!"
```

- L'exemple Windows cURL suivant publie un message persistant dans la chaîne de rubrique myTopic/thisTopic sur le gestionnaire de files d'attente QM1, avec une expiration de 2 minutes. Le message contient le texte *"Hello World !"*:

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/topic/myTopic%2FthisTopic/
message"
-X POST -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token_value"
-H "Content-Type: text/plain;charset=utf-8" -H "ibm-mq-md-persistence: persistent"
-H "ibm-mq-md-expiry: 120000" --data "Hello World!"
```





## Remarques

---

:NONE.

Le présent document peut contenir des informations ou des références concernant certains produits, logiciels ou services IBM non annoncés dans ce pays. Consultez votre représentant IBM local pour obtenir des informations sur les produits et services actuellement disponibles dans votre région. Toute référence à un produit, logiciel ou service IBM n'implique pas que seul ce produit, logiciel ou service IBM puisse être utilisé. Tout produit, programme ou service fonctionnellement équivalent qui ne porte pas atteinte à un droit de propriété intellectuelle IBM peut être utilisé à la place. Il est de la responsabilité de l'utilisateur d'évaluer et de vérifier lui-même les installations et applications réalisées avec des produits, logiciels ou services non expressément référencés par IBM.

IBM peut détenir des brevets ou des demandes de brevet couvrant les produits mentionnés dans le présent document. La remise de ce document ne vous donne aucun droit de licence sur ces brevets ou demandes de brevet. Si vous désirez recevoir des informations concernant l'acquisition de licences, veuillez en faire la demande par écrit à l'adresse suivante :

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785  
U.S.A.

Pour obtenir des informations sur les licences relatives aux informations sur deux octets (DBCS), contactez le service de la propriété intellectuelle IBM de votre pays ou envoyez vos demandes de renseignements, par écrit, à :

Intellectual Property Licensing  
Legal and Intellectual Property Law  
IBM Japan, Ltd.  
19-21, Nihonbashi-Hakozakicho, Chuo-ku  
Tokyo 103-8510, Japan

**Le paragraphe suivant ne s'applique ni au Royaume-Uni, ni dans aucun pays dans lequel il serait contraire aux lois locales.** LE PRESENT DOCUMENT EST LIVRE "EN L'ETAT" SANS AUCUNE GARANTIE EXPLICITE OU IMPLICITE. IBM DECLINE NOTAMMENT TOUTE RESPONSABILITE RELATIVE A CES INFORMATIONS EN CAS DE CONTREFACON AINSI QU'EN CAS DE DEFAUT D'APTITUDE A L'EXECUTION D'UN TRAVAIL DONNE. Certaines juridictions n'autorisent pas l'exclusion des garanties implicites, auquel cas l'exclusion ci-dessus ne vous sera pas applicable.

Le présent document peut contenir des inexactitudes ou des coquilles. Ce document est mis à jour périodiquement. Chaque nouvelle édition inclut les mises à jour. IBM peut, à tout moment et sans préavis, modifier les produits et/ou programmes décrits dans ce document.

Les références à des sites Web non IBM sont fournies à titre d'information uniquement et n'impliquent en aucun cas une adhésion aux données qu'ils contiennent. Les éléments figurant sur ces sites Web ne font pas partie des éléments du présent produit IBM et l'utilisation de ces sites relève de votre seule responsabilité.

IBM pourra utiliser ou diffuser, de toute manière qu'elle jugera appropriée et sans aucune obligation de sa part, tout ou partie des informations qui lui seront fournies.

Les licenciés souhaitant obtenir des informations permettant : (i) l'échange des données entre des logiciels créés de façon indépendante et d'autres logiciels (dont celui-ci), et (ii) l'utilisation mutuelle des données ainsi échangées, doivent adresser leur demande à :

IBM Corporation  
Coordinateur d'interopérabilité logicielle, département 49XA

3605 Autoroute 52 N  
Rochester, MN 55901  
U.S.A.

Ces informations peuvent être soumises à des conditions particulières, prévoyant notamment le paiement d'une redevance.

Le logiciel sous licence décrit dans le présent document et tous les éléments sous disponibles s'y rapportant sont fournis par IBM conformément aux dispositions du Contrat sur les produits et services IBM, aux Conditions Internationales d'Utilisation de Logiciels IBM ou de tout autre accord équivalent.

Les données de performance indiquées dans ce document ont été déterminées dans un environnement contrôlé. Par conséquent, les résultats peuvent varier de manière significative selon l'environnement d'exploitation utilisé. Certaines mesures évaluées sur des systèmes en cours de développement ne sont pas garanties sur tous les systèmes disponibles. En outre, elles peuvent résulter d'extrapolations. Les résultats peuvent donc varier. Il incombe aux utilisateurs de ce document de vérifier si ces données sont applicables à leur environnement d'exploitation.

Les informations concernant des produits non IBM ont été obtenues auprès des fournisseurs de ces produits, par l'intermédiaire d'annonces publiques ou via d'autres sources disponibles. IBM n'a pas testé ces produits et ne peut confirmer l'exactitude de leurs performances ni leur compatibilité. Elle ne peut recevoir aucune réclamation concernant des produits non IBM. Toute question concernant les performances de produits non IBM doit être adressée aux fournisseurs de ces produits.

Toute instruction relative aux intentions d'IBM pour ses opérations à venir est susceptible d'être modifiée ou annulée sans préavis, et doit être considérée uniquement comme un objectif.

Le présent document peut contenir des exemples de données et de rapports utilisés couramment dans l'environnement professionnel. Ces exemples mentionnent des noms fictifs de personnes, de sociétés, de marques ou de produits à des fins illustratives ou explicatives uniquement. Toute ressemblance avec des noms de personnes, de sociétés ou des données réelles serait purement fortuite.

Licence sur les droits d'auteur :

Le présent logiciel contient des exemples de programmes d'application en langage source destinés à illustrer les techniques de programmation sur différentes plateformes d'exploitation. Vous avez le droit de copier, de modifier et de distribuer ces exemples de programmes sous quelque forme que ce soit et sans paiement d'aucune redevance à IBM, à des fins de développement, d'utilisation, de vente ou de distribution de programmes d'application conformes aux interfaces de programmation des plateformes pour lesquels ils ont été écrits ou aux interfaces de programmation IBM. Ces exemples de programmes n'ont pas été rigoureusement testés dans toutes les conditions. Par conséquent, IBM ne peut garantir expressément ou implicitement la fiabilité, la maintenabilité ou le fonctionnement de ces programmes.

Si vous visualisez ces informations en ligne, il se peut que les photographies et illustrations en couleur n'apparaissent pas à l'écran.

## Documentation sur l'interface de programmation

---

Les informations d'interface de programmation, si elles sont fournies, sont destinées à vous aider à créer un logiciel d'application à utiliser avec ce programme.

Ce manuel contient des informations sur les interfaces de programmation prévues qui permettent au client d'écrire des programmes pour obtenir les services de WebSphere MQ.

Toutefois, lesdites informations peuvent également contenir des données de diagnostic, de modification et d'optimisation. Ces données vous permettent de déboguer votre application.

**Important :** N'utilisez pas ces informations de diagnostic, de modification et d'optimisation en tant qu'interface de programmation car elles sont susceptibles d'être modifiées.

## Marques

---

IBM, le logo IBM , ibm.com, sont des marques d' IBM Corporation dans de nombreux pays. La liste actualisée de toutes les marques d' IBM est disponible sur la page Web "Copyright and trademark information" [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml). Les autres noms de produits et de services peuvent être des marques d'IBM ou d'autres sociétés.

Microsoft et Windows sont des marques de Microsoft Corporation aux Etats-Unis et/ou dans certains autres pays.

UNIX est une marque de The Open Group aux Etats-Unis et dans certains autres pays.

Linux est une marque de Linus Torvalds aux Etats-Unis et/ou dans certains autres pays.

Ce produit inclut des logiciels développés par le projet Eclipse (<https://www.eclipse.org/>).

Java ainsi que tous les logos et toutes les marques incluant Java sont des marques d'Oracle et/ou de ses sociétés affiliées.







Référence :

(1P) P/N: